Universitat de Vic
Escola Politècnica Superior

Treball Final de Màster

Pots&Plants

Aplicació de Realitat Augmentada

Jesús Mosquera Mayo

Màster Apps & Games

Directora: Sergi Grau

Vic, setembre de 2014

POTS AND PLANTS

Índex

1. INTRODUCCIÓ I CONCEPTUALITZACIÓ 1
1.1. Introducció 2
1.2. Idea 2
1.3. Descripció 2
1.4. Objectius 3
1.5. Metodologia 4
2. ESTAT DEL DESENVOLUPAMENT 5
2.1. Marc teòric 6
2.1.1. Realitat Augmentada 6
2.1.2. Aplicacions de Realitat Augmentada 9
2.1.3 Apps de geo-posicionament 9
2.1.4 Apps de reconeixement 10
2.2. Benchmarking 11
2.2.1. Criteris 11
2.2.2. Taules 12
2.3. Tecnologia 13
2.3.1. Unity 13
2.3.2. Vuforia 15
2.3.3. Blender 16
3. JOC I MECANICA 18
3.1. Concepte del joc 19
3.2. Característiques principals 19
3.3. Gènere i tipus de joc 20
3.4. Definició del públic objectiu 20
4. ARQUITECTURA DE LA INFORMACIÓ 21
4.1. Game Flow 22
4.2. Diagrames de flux 23
4.2.1. Diagrama de flux general 23
4.2.2. Diagrama de flux de pantalla de joc 24
4.2.3. Diagrama de flux del joc de terra 25
4.2.4. Diagrama de flux del joc d'aigua 27
4.2.5. Diagrama de flux del joc de llum 28
5. DISSENY DE LA INTERFICIE 29
5.1. Identitat corporativa 29
5.1.1. El logotip 29

5.1.2. Metàfora gràfica 29
5.1.3. Colors corporatius 30
5.1.4. Tipografia 31

6. HISTORIA I ENTORN 33
6.1. Història i narrativa 34
6.1.1. Història de fons 34
6.2.1. Elements de la trama 34

7. DISSENY DE PANTALLES I PERSONATGES 39
7.1. Disseny de pantalles 40
7.1.1. Pantalla de benvinguda 40
7.1.2. Pantalla de nivells 40
7.1.3. Pantalla de joc 41
7.1.4. Pantalla d'ajuda 42
7.2. Disseny de personatges 43
7.2. Disseny de marcadors 45

8. CONCLUSIONS 47
8.1. Generales i/o especifiques 48
8.2. Futures implementacions 48
8.3. Valoració del resultat en relació als objectius plantejats 49

APÈNDIX 50
A.1. Apèndix Jocs. 51
4.1.1. Recettear an ítem Shop's Tale 51
4.1.2. Carcassone 55
4.1.3. Archer - Nintendo DS: jocs RA. 59
4.1.4. Fishing - Nintendo DS: jocs RA. 61
4.1.5. Billiard - Nintendo DS: jocs RA. 62
A.2. Apèndix links. 64
A.3. Apèndix Codi. 66

Figures i taules

Figura 1 – Pots&Plants Pantalla de nivells..... 3
Figura 2 – Sensorama..... 6
Figura 3 – Videoplace..... 7
Figura.4 – De Món real a Món Virtual..... 7
Figura 5 – ARToolKit..... 8
Figura 6 – Wikitude..... 9
Figura 7 – marcador AR..... 10
Taula 1 – Estudi de mercat.....12
Figura 8 – framework Unity3D..... 13
Figura.9 – MonoDevelop-Unity..... 14
Figura 10 – imatge target dins de Unity3D..... 15
Figura 11 – Vuforia Target Manager…….. 16
Figura 12 – Blender……..…… 17
Figura 13 – Proves de logotip……..…… 29
Figura 14 – Colors corporatius……..…….. 30
Figura 15 – Tipografia Kingthings Annex …….. 31
Figura 16 – Tipografia Thin Cool……..…….. 31
Figura 17 – Tipografia Source Sans Pro……... 32
Figura 18 – icona de terra de Pots&Plants……... 34
Figura 19 – Level 1 Pots&Plants…… 35
Figura 20 – Earth tutorial de Pots&Plants…….. 36
Figura 23 – Water tutorial de Pots&Plants…… 37
Figura 24 – icona de la llum de Pots&Plants…… 37
Figura 25 – Level 2 de Pots&Plants……... 38
Figura 26 – Light tutorial de Pots&Plants…….. 38
Figura 27 – Pantalla de benvinguda……... 40
Figura 28 – Pantalla de nivells…… 41
Figura 29 – Pantalla de joc…….. 42

Figura 30 – Pantalla d'ajuda..... 43
Figura 31 – Representacions de Test..... 43
Figura 32 – Representacions de Margarita..... 44
Figura 33 – Representacions de Cactus..... 44
Figura 34 – marcador principal de Pots&Plants..... 45
Figura 35 – marcador secundari de Pots&Plants..... 46
Figura 36 – Recettear..... 51
Figura 37 – masmorra de Recettear..... 52
Figura 38 – missatges de Recettear..... 53
Figura 39 – comandes de Recettear..... 54
Figura 40 – marcador de 3 jugadors..... 55
Figura 41 – Col·locació parcel.la..... 55
Figura 42 – Segments de parcel·les..... 56
Figura 43 – Seguidors al taulell..... 56
Figura 44 – Punts per camí. 57
Figura 45 – Punts per ciutat. 57
Figura 46 – Claustre complert. 58
Figura 47 – Punts per granges..... 58
Figura 48 – Nintendo DS..... 59
Figura 49 – Nintendo Cards..... 60
Figura 50 – Archer..... 60
Figura 51 – Drac Archer..... 61
Figura 52 – Fishing..... 62
Figura 53 – Billiards..... 63
Figura 54 – Drac Billiards..... 63

POTS AND PLANTS

1. INTRODUCCIÓ I CONCEPTUALITZACIÓ

1.1. Introducció

Estic interessat en fer que el meu treball final de màster sigui una aplicació de Realitat Augmentada. Crec que es tracta d'una tecnologia emergent amb moltes perspectives de futur.

Crec això, degut a que dintre del món de R + D es parla de pantalles de grafè o les tan esperades Google Glass. No cal dir que són dispositius que van lligats amb la Realitat Augmentada.

Es per això que m'agradaria fer una exploració d'aquesta tecnologia. On l'usuari pot experimentar les diverses formes d'interactivitat que disposem.

1.2. Idea

Joc de Realitat Augmentada on l'usuari haurà de complir petits reptes interactuant amb els elements virtuals de l'escena. Aquests elements es presentaran fent us de marcadors. Per tant aquest projecte es centra en el desenvolupament d'una aplicació de RA d'escriptori tal i com es pot llegir més endavant.

1.3. Descripció

El projecte és un joc on l'usuari ha de cuidar unes plantes. Per a poder fer aquesta feina el jugador realitzarà 3 tipus de reptes. Aquests reptes són petits jocs, és a dir, que hi ha tres tipus de "mini-jocs" dintre de l'Aplicació. Degut a que cada jugador té preferències diferents, aquesta divisió de jocs permet accedir a un major nombre d'usuaris.

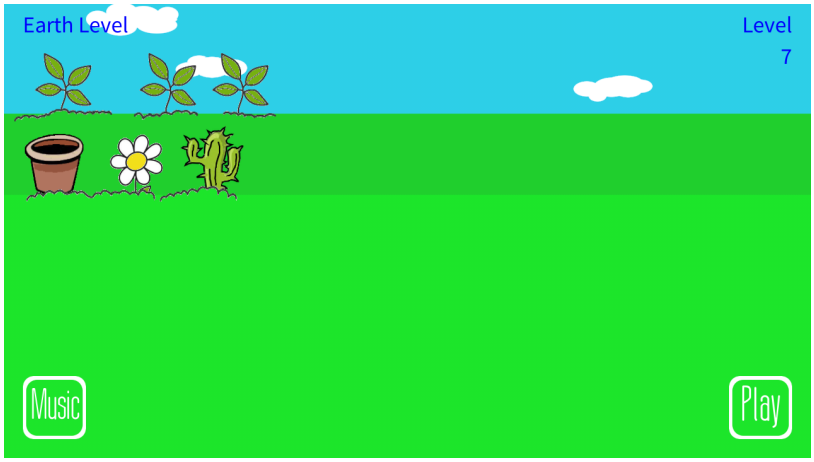


Figura 1 – Pots&Plants pantalla de nivells

Al primer repte, s'haurà de tenir cura de la terra de la planta. Aquest mini-joc serà un joc de memòria on l'usuari ha de fer la composició de la terra que tindrà la planta.

Al segon repte, li donarem a la planta l'aigua que necessita. L'usuari es troba davant d'un joc de dispars.

Al tercer repte, hem de fer que li doni llum a la nostra planta. En aquest s'ha de complir un repte de laberint.

1.4. Objectius

- Principal : Crear un joc de Realitat Augmentada.
- Secundari: Apendrer el funcionament de Vuforia.
- Secundari: Millorar mètodes de programació amb llenguatges orientats a objectes.

1.5. Metodologia

1. Recollida de informació:

    a. Recull d'informació i evolució històrica de la Realitat Augmentada així com distinció i tipus d'aplicacions.
    b. Agafant referents de jocs similars en el mercat: PC, Apps i vídeoconsoles. Com a base d'inspiració per a la creació de la historia del joc.
    c. Recollida de requeriments tècnics per al desenvolupament tecnològic a nivell de programació i disseny.

2. Anàlisis de la informació recollida i fer un primer document del que creiem que ha de tenir la nostre aplicació (requeriments funcionals i tècnics de l'aplicació)..

3. Disseny de la nostre aplicació: història, personatges, pantalles, flux del joc.

4. Desenvolupament del codi.

5. Test intern i correcció a la mancança de falles de l'historia. Errors de bugs del codi, etc...

6. Per a desplegar el programa utilitzaríem les xarxes socials, com a canal de màrqueting i per la seva venta utilitzaríem els Markets de les plataformes en que s'ha desenvolupat.

2. ESTAT DEL DESENVOLUPAMENT

2.1. Marc teòric

2.1.1. Realitat Augmentada

A la Realitat Augmentada o RA a diferència de la Realitat Virtual el món real no és substituït per un món fictici, sinó que s'aprofiten tots els elements que envolten la realitat física per afegir-hi continguts virtuals.

Per l'usuari de RA la informació virtual que va rebent es combina de forma simultània amb el món real que l'envolta. On aquests elements virtuals poden ser interactius i manipulables.

Per tant podríem afirmar que l'objectiu de la RA és ampliar la informació del nostre entorn que estem observant en un moment donat i que l'usuari pugui interactuar amb aquesta informació i fins i tot decidir el seu contingut.

Breu cronologia:

- **Anys 30**: apareix el concepte de Realitat Virtual.

- **1962**: Morton Heilig crea el primer simulador de moto anomenat Sensorama. Aquest simulador combina en el mateix temps i espai, imatges, sons, vibracions i l'olfacte.

Figura 2 – Sensorama

- **1973**: Ivan Sutherland inventa el display de cap (HMD) una finestra a un món virtual.

- **1985**: Myron Krueger crea Videoplace que permet als usuaris interactuar amb objectes virtuals per primera vegada.

Figura 3 – Videopalce

- **1990**: Jaron Lanier crea el terme Realitat Virtual i genera la primera activitat comercial al voltat dels mons virtuals.

- **1992**: Tom Caudell crea el terme Realita Augmentada.

- **1994**: Steven Feiner, Blair MacIntyre i Doree Seligmann primera utilització important d'un sistema de Realitat Augmentada en un prototip, KARMA, presentat en la conferència de la interfície gràfica.

- **1994**: Paul Milgram i Fumio Kishino defineixen la Realitat Augmentada com una zona intermèdia en un espai fictici i continu que uneix el món real i el món virtual.

Figura 4 – De Món real a Món Virtual

- **1997**: Ronald Azuma va definir el sistema de Realitat Augmentada com, aquell sistema que combina elements reals i virtuals, és interactiu en temps real i és capaç d'enregistrar en 3D.

- **1997**: Neix l'Smartphone, aparell que a dia d'avui s'ha convertit en la plataforma de Realitat Augmentada pel gran públic.

- **1999**: Hirokazu Kato desenvolupa ARToolKit per C++ (llibreria open-source per a la creació d'aplicacions de Realitat Augmentada), que és una de les principals llibreries de desenvolupament d'AR.

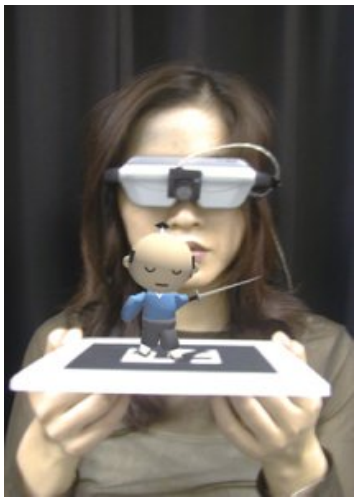


Figura 5 – ARToolKit

- **2001**: Apareixen les primeres aplicacions de Realitat Augmentada que fan ús del GPS per a detectar la posició del receptor.

- **2006**: Apareixen les primeres aplicacions per a mòbils de Realitat Augmentada que fan servir geolocalització, acceleròmetres, la càmera del dispositiu i la pantalla.

- **2010**: Surt al mercat Invizimals, el primer joc per a una consola que fa ús de la Realitat Augmentada.

Amb la millora de la tecnologia ha anat augmentat el nombre d'aplicacions amb aquest tipus de tecnologia ja que paral·lelament han anat millorant en quan a qualitat i processament, fent més atractives a l'usuari final.

Com a tecnologia jove i emergent són molt els reptes que li queden a la Realitat Augmentada i de fet en l'actualitat hi ha un gran nombre d'investigacions obertes que

es relacionen amb la seva millora: el reconeixement d'imatges, la coordinació entre els elements reals i virtuals, nous dispositius, la reducció d'interferències als sensors d'orientació, nous materials, etc...

Malgrat tot i aquests reptes, que portin a pensar que encara ens trobem davant una tecnologia incipient i poc fiable, són moltes les possibilitats que en l'actualitat la RA ens ofereix i hi ha un camí molt gran a recorre en el desenvolupament d'aplicacions amb aquest tipus de tecnologies en camps tant diversos com: l'entreteniment, la medicina, la publicitat, arquitectura, el turisme, simulació, ...

2.1.2. Aplicacions de Realitat Augmentada

Quan es parla de Realitat Augmentada hem de tenir en compte els diferents tipus d'aplicacions que ens podem trobar. Es pot distingir clarament dos vessants diferents. Per un cantó hi ha aplicacions centrades en la geo-localització i també aplicacions centrades en reconeixement d'objectes i/o marcadors.

2.1.3 Apps de geo-posicionament

Les aplicacions de Realitat Augmentada que fan ús de la localització del dispositiu són les més esteses. Parlem d'aplicacions que fan servir el maquinari del nostre dispositiu per a localitzar i donar-nos informació sobre punts d'interès que es mostren a la pantalla del mòbil.

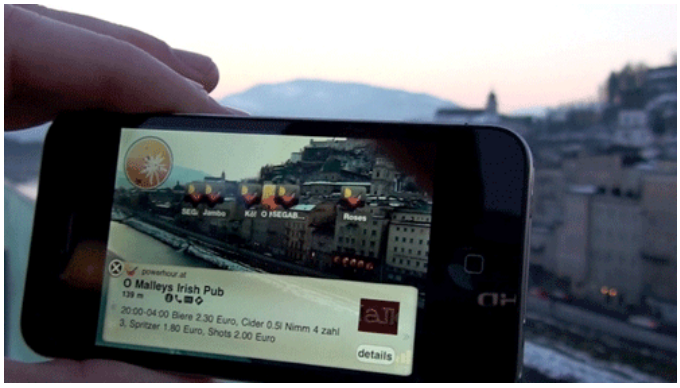


Figura 6 – Wikitude

Dintre d'aquests tipus d'aplicacions ens trobem exemples com Layar i Wikitude.

Fent-ne us d'una aplicació podem conèixer on es troba la farmàcia més propera, quines cases troben a la venda o tenir en detall la informació històrica de l'edifici que tenim davant.

2.1.4 Apps de reconeixement

Dintre de les aplicacions de reconeixement hi ha d'aplicacions que requereixen l'ús de marcadors on es superposa la informació virtual ja siguin models 3D, vídeos, etc... Es necessari per això un programari específic que reconegui el marcador i ens mostri aquesta informació. Cal dir que aquests marcadors també poden ser fotografies.

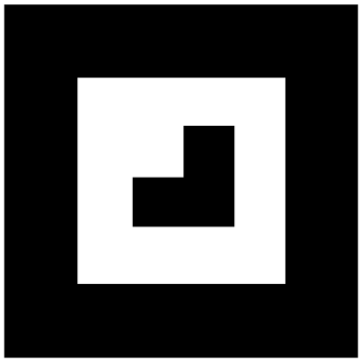


Figura 7 – marcador AR

El funcionament d'aquestes aplicacions segueix un procés on hem d'imprimir-nos el marcador, obrir l'aplicació i aconseguir que el marcador estigui dintre de la zona visible de la càmera en aquest moment tindrem la informació desitjada. Generalment parlem de models 3D.

2.2. Benchmarking

2.2.1. Criteris

Per a l'anàlisi del Benchmarking s'han seguit criteris de recerca en funció de 3 conceptes: aplicacions de realitat augmentada, jocs casuals i jocs generals.

La recerca de jocs generals s'ha fet per a agafar idees o conceptes que puguin afegir-se a la mecànica del joc donant més cós a la jugabilitat com procés de gestió dels jocs.

Jocs casuals, parlem de jocs curts i petits.

2.2.2. Taules

Nom	URL	Breu descripció
Recettear	http://en.wikipedia.org/wiki/Recettear:_An_Item_Shop's_Tale	Joc casual que combina l'estil RPG i de comerç on la nostra finalitat es aconseguir assolir el deute de la nostra botiga.
Carcassone	http://ca.wikipedia.org/wiki/Carcassonne_(joc)	Joc de taula on s'ha de crear un mapa competint amb fer-se el màxim nombre de punts agafant les millors distribucions de recursos.
Archer	https://www.youtube.com/watch?v=IKS0caNPR4o	Joc de RA de la vídeo consola Nintendo DS on s'han de disparar fletxes a dianes i enemics.
Fishing	http://www.wired.com/2011/02/nintendo-3ds-ar-games	Joc de pesca i RA de la vídeo consola Nintendo DS. L'objectiu del joc es col·leccionar peixos fins a un màxim de 31 tipus.
Billiards	http://www.cubed3.com/review/950/1/ar-games-ar-shot-nintendo-3ds.html	Joc de RA de la vídeo consola Nintendo DS on s'ha d' aconseguir arribar al forat amb la bola amb mínim nombre de cops

Taula 1 – Estudi de mercat

2.3. Tecnologia

2.3.1. Unity

Per la realització del projecte cal un entorn, o framework, específic de treball. L'entorn de desenvolupament que es fa servir és Unity.

Unity 3d és una eina per a desenvolupament de videojocs que permet crear videojocs. A més és una eina completament multi-plataforma, es pot desenvolupar per pc, mac, web, iOS, Android, Wii, Xbox i PS.

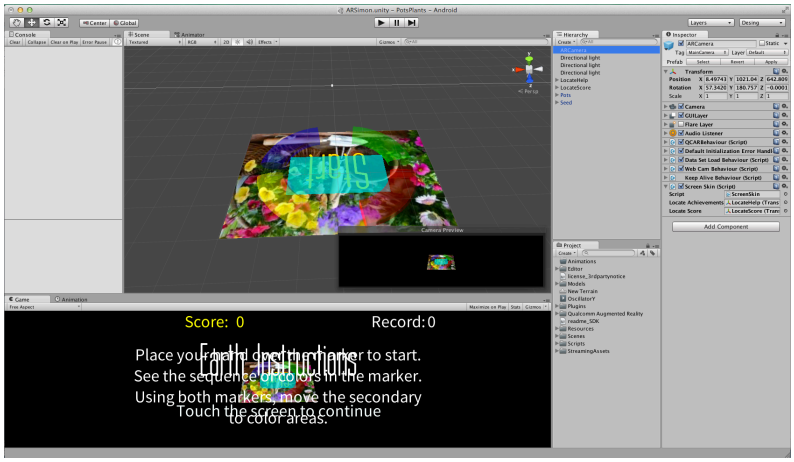


Figura 8 – framework Unity3D

Tots els recursos 3d i les textures que s'utilitzen a Unity han de crear-se amb programes externs de disseny. Unity és capaç d'importar fàcilment recursos de Maya, Cinema 4D, 3ds Max, Cheetah3D i Blender. A més a més permet importar formats .fbx i .obj. Dels editors anteriors Blender és l'únic Open-Source.

Pel que fa al codi, Unity permet l'ús de tres llenguatges: C#, Boo i JavaScript. El JavaScript que fa servir Unity és una versió pròpia. En el cas del projecte es fa tot amb C#.

Tota la lògica del joc a Unity està basada en Mono, una plataforma .net per crear aplicacions cross-platform.

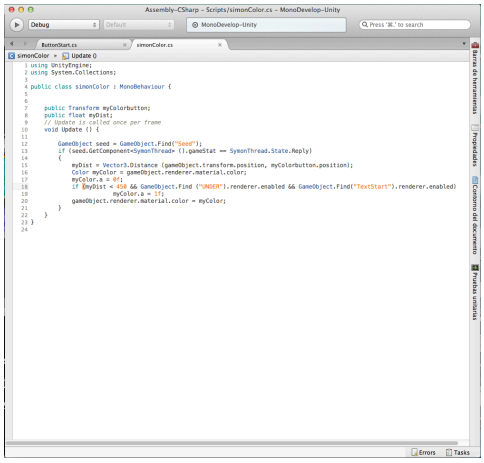


Figura 9 – MonoDevelop-Unity

Tot script, petit programa, deriva de la classe MonoBehaviour que té cinc funcions bàsiques que controlen quan s'executa el codi, a cada frame o l'inicialitzar-se. Les funcions són:

- Update: s'executa una vegada en cada frame.
- LateUpdate: també s'executa després de la funció Update.
- FixedUpdate: es crida a cada cicle del motor de físiques, en aquesta funció han de posar-se els càlculs que afectin al motor.
- Awake: es crida quan l'script es carrega en temps d'execució i és un bon lloc per posar inicialitzacions i referències entre scripts.
- Start: es crida després de la funció Awake i abans del primer Update, és un altre bon lloc on posar inicialitzacions o comprovacions que només vagin a fer-se un cop.

2.3.2. Vuforia

Vuforia de Qualcomm és una llibreria que permet desenvolupar aplicacions de Realitat Augmentada per a dispositius mòbils amb iOS o Android. Aquesta llibreria es pot importar a Unity3d de forma senzilla.

El funcionament es basa en la detecció de marcadors específics amb la càmera del dispositiu. Les funcions de la llibreria proporcionen la posició i orientació dels marcadors a través d'una matriu.

Una vegada que la llibreria reporta la detecció del marcador al món ens dóna la seva matriu. Aquesta informació es tracta habitualment per col·locar un model 3D a la posició i amb l'orientació indicada per la matriu.

Vuforia permet treballar amb tres tipus de imatges o marcadors diferents: image targets, multi targets i frame markers. Al projecte s'ha treballat amb les imatge targets.

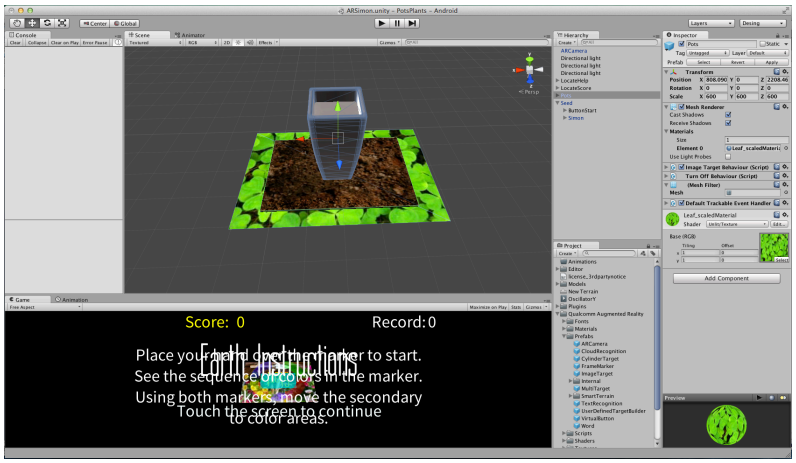


Figura 10 – imatge target dins de Unity3D

Image targets són imatges amb la complexitat necessària perquè la llibreria de Vuforia pugui trobar suficients referències dins de la mateixa i així poder detectar i tractar-la com a patró.

Aquestes imatges, un cop localitzades per primera vegada permeten a la llibreria seguir detectant la seva posició encara que estiguin només parcialment cobertes i enfocades per la càmera.

Una altre característica interessant és que es poden utilitzar imatges seleccionades pel programador entre les imatges que es volen detectar. Per a això s'utilitza una eina en línia que es troba a la web de Vuforia.

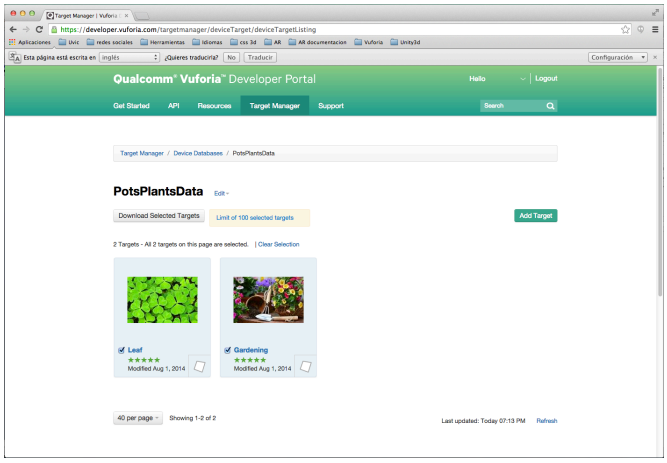


Figura 11 – Vuforia Target Manager

2.3.3. Blender

Blender és un programari lliure multiplataforma, dedicat a l'edició tridimensional. Això inclou eines per al modelatge d'objectes, l'edició de materials, l'animació, entre d'altres coses.

El programa fou inicialment distribuït de forma gratuïta però sense el codi font. Actualment és compatible amb totes les versions de Microsoft Windows, GNU/Linux, Solaris, FreeBSD, IRIX i Mac OS X.

Figura 12 – Blender

Tenint en compte el fet que sigui multiplataforma, lliure, gratuït i amb un mida d'origen molt petita; li dona els requeriments perfectes per a ser una eina de treball més dintre d'aquest projecte.

A més a més, com ja s'ha dit, Unity3d accepta sense problemes els models generats amb Blender.

3. JOC I MECANICA

3.1. Concepte del joc

Joc en primera persona a on el jugador a d'intentar mantenir la cura de les seves plantes mitjançant petits reptes representats per cadascun dels mini-jocs.

Dintre de la divisió de jocs tenim el primer joc: un joc de memòria estil Simon**.**

A un dels marcadors podrem veure un disc dividit en 4 sectors. Cada sector té un color: vermell, verd, blau i groc.

El joc aleatòriament va il·luminant els diferents quadrants de colors mentre que a la vegada es reprodueix un so. Després de veure la seqüencia l'usuari ha de reproduir-la amb el mateix ordre.

El segon joc ens trobem un joc de disparar. L'usuari es troba amb un núvol donant voltes a la planta.

Cada cop que el jugador toca la pantalla dispara una boleta que va cap endavant i al centre de la pantalla. Si colpeja al núvol aquest es destrueix. El nou núvol tindrà una velocitat de gir i velocitat centrífuga diferent així com un radi inicial diferent.

Al tercer joc l'usuari es troba amb un marcador que té un laberint. Aquest laberint sempre serà de la mida del marcador però es genera aleatòriament. En aquest joc la idea es fer servir la gravetat per desplaçar una boleta fins al forat que es troba al costat contrari del laberint.

3.2. Característiques principals

- El jugador juga en primera persona.
- Joc individual.
- Simulació mitjançant Realitat Augmentada.

3.3. Gènere i tipus de joc

El joc té tres tipus de mini-jocs separats completament entre ells respecte a concepte, això és degut a que cada persona té unes preferències diferents.

Atenent a l'explicació de l'apartat de concepte de joc, la classificació es pot fer de la següent manera:

- Joc de terra es pot classificar com un joc de memòria o un joc estil Simon.
- Joc d'aigua es pot classificar com un joc de dispars.
- Joc de llum es pot classificar com un joc de laberint.

3.4. Definició del públic objectiu

L'Aplicació és un joc per a un jugador que experimenta la Realitat Augmentada.

Dintre de PEGUI trobem la classificació +3 ajustada a l'Aplicació d'aquest projecte, ja que no conté cap tipus d'escena violenta, por, sexe, llenguatge obscè, etc...

PEGI o Pan European Game Information (en català: informació de videojocs a la zona europea) és el sistema europeu per classificar el contingut dels videojocs i altres tipus de programari d'entreteniment.

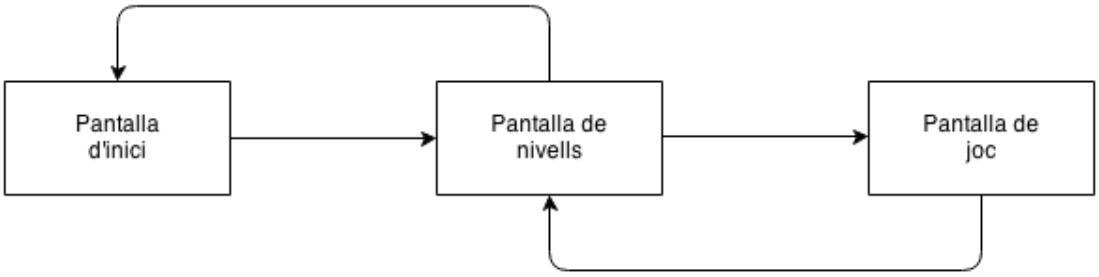
De totes maneres l'usuari ha de fer servir marcadors impresos. Així doncs aquest joc requereix que l'usuari tingui com a mínim 5 anys per poder imprimir els marcadors. Ja que amb aquesta edat té els coneixements per realitzar aquesta tasca.

Per altra banda l'objectiu és experimentar la interactivitat que ens pot oferir la Realitat Augmentada. És per aquest motiu que l'edat màxima dels usuaris no està acotada.

Es pot establir doncs que el joc va destinat a persones mes grans de 5 anys.

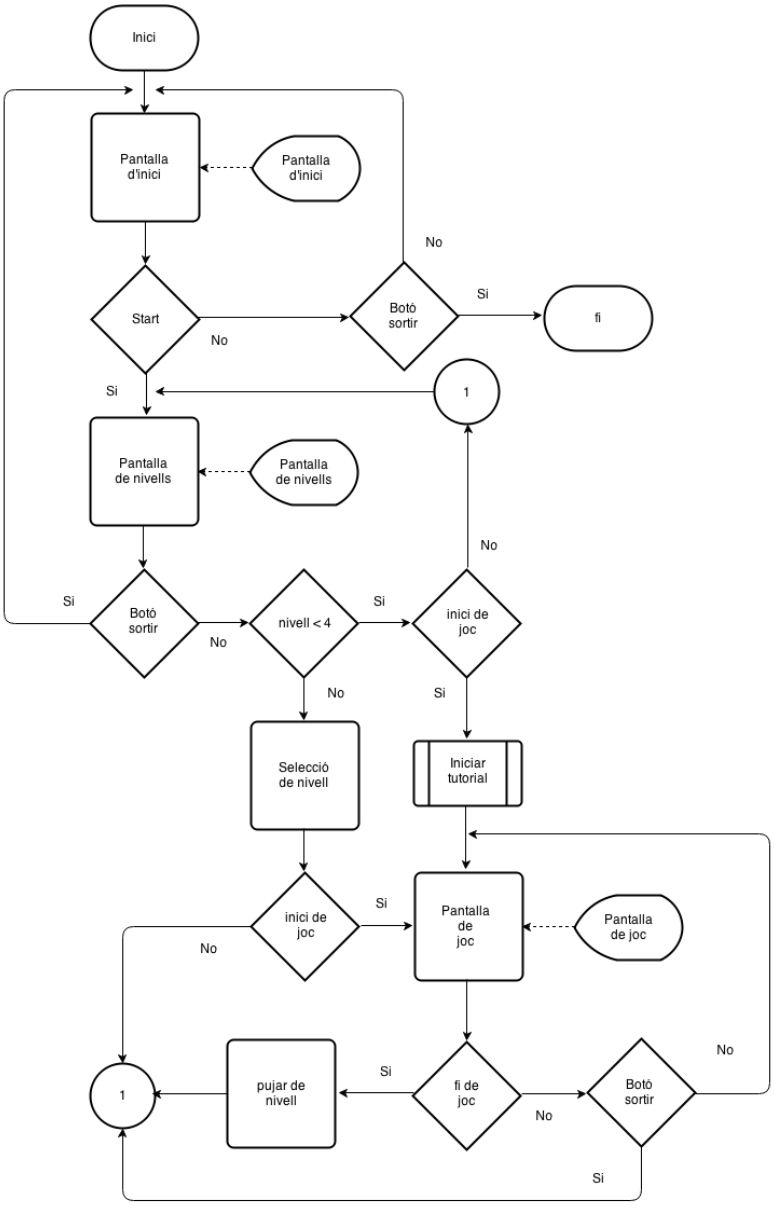
4. ARQUITECTURA DE LA INFORMACIÓ

4.1. Game Flow

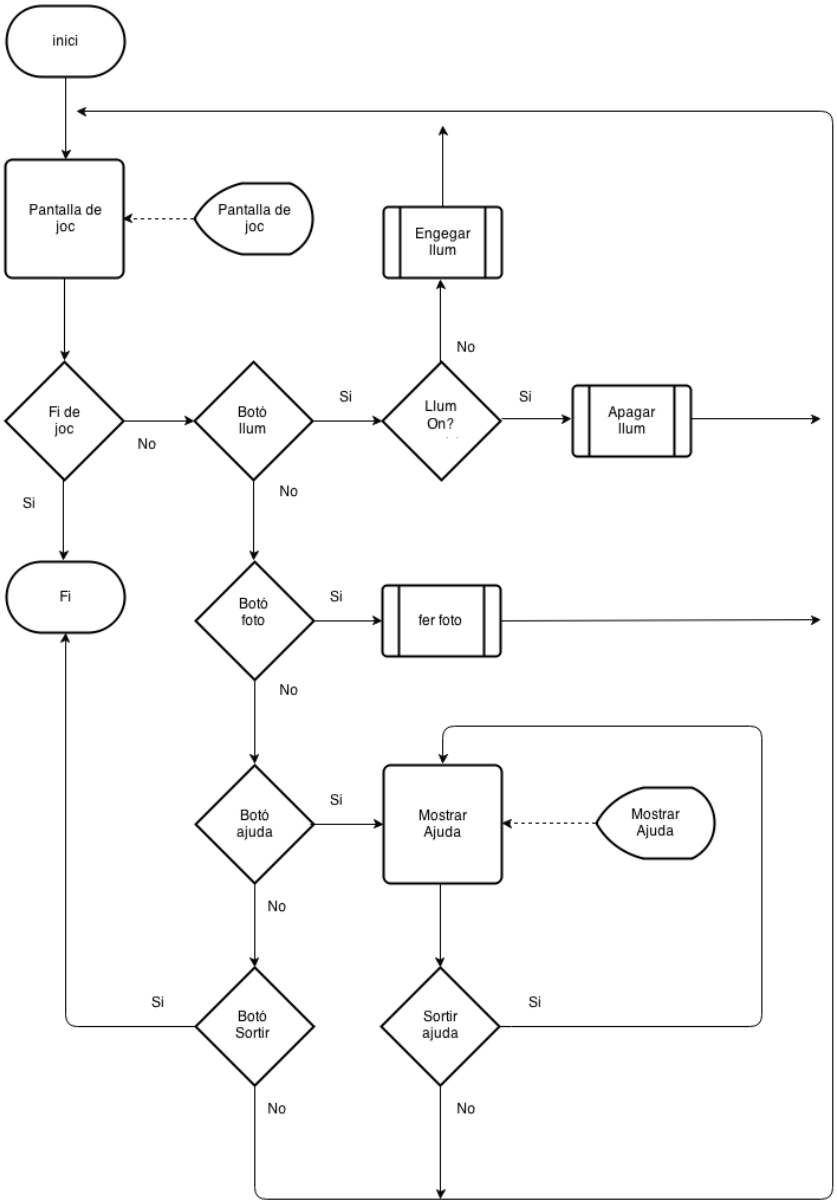


4.2. Diagrames de flux

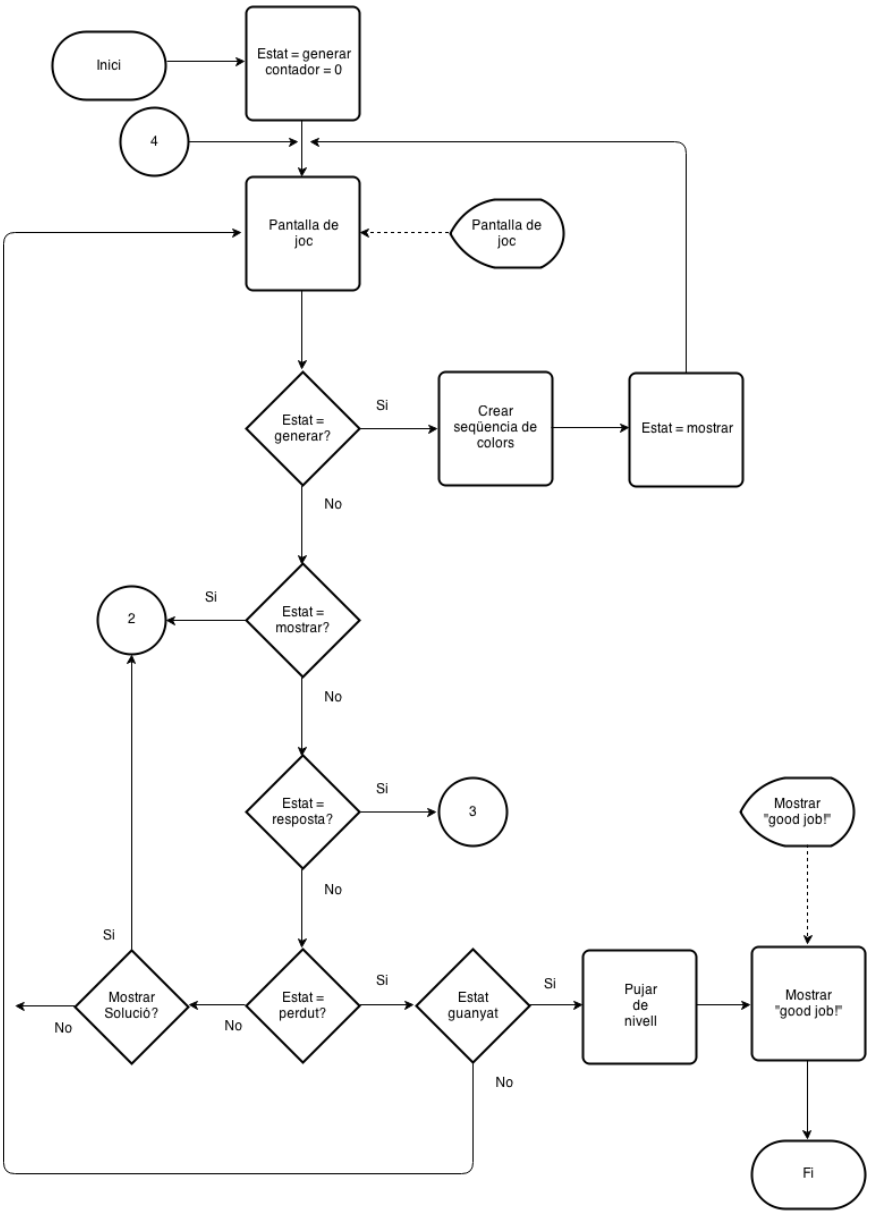
4.2.1. Diagrama de flux general

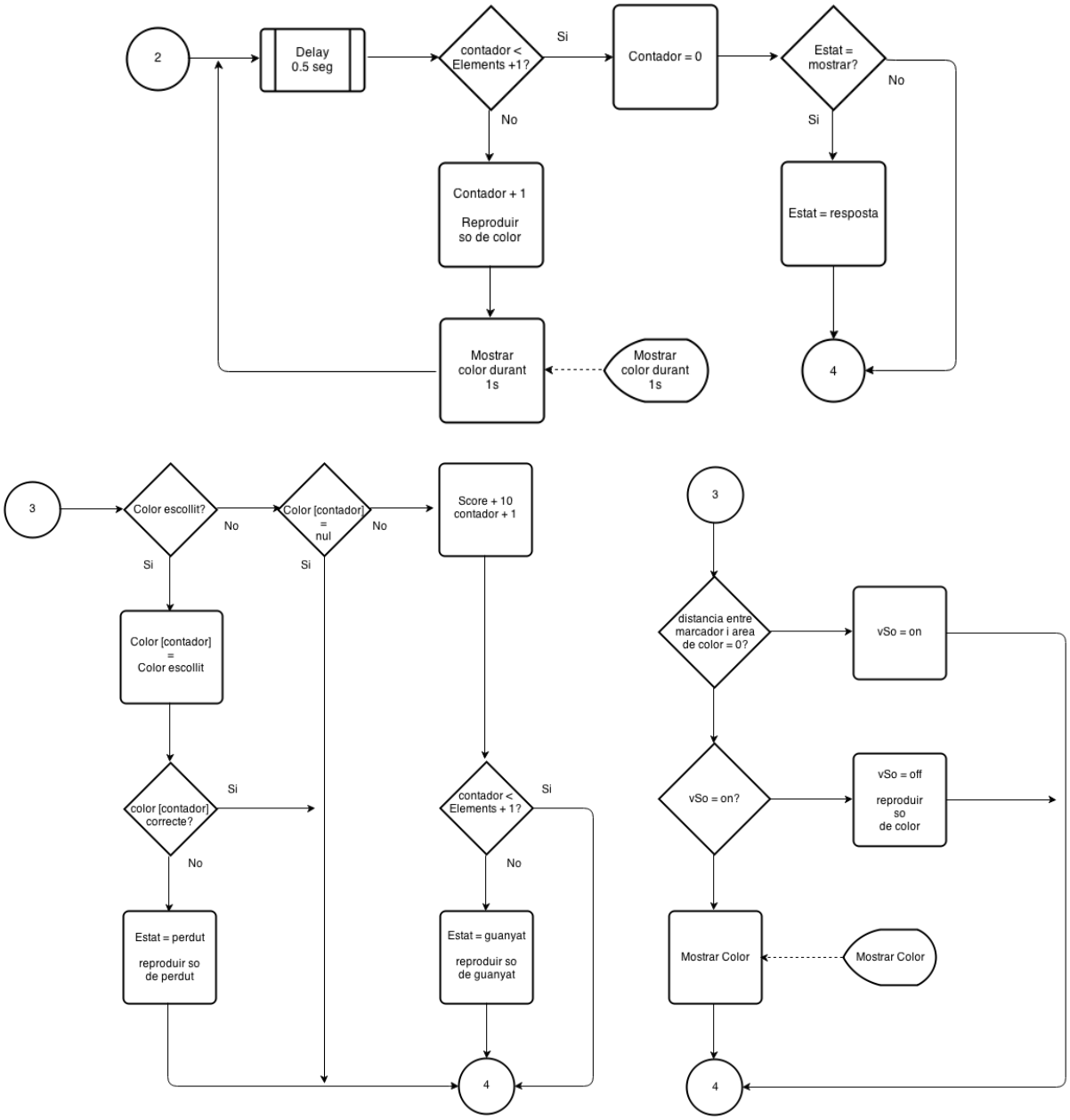


4.2.2. Diagrama de flux de pantalla de joc

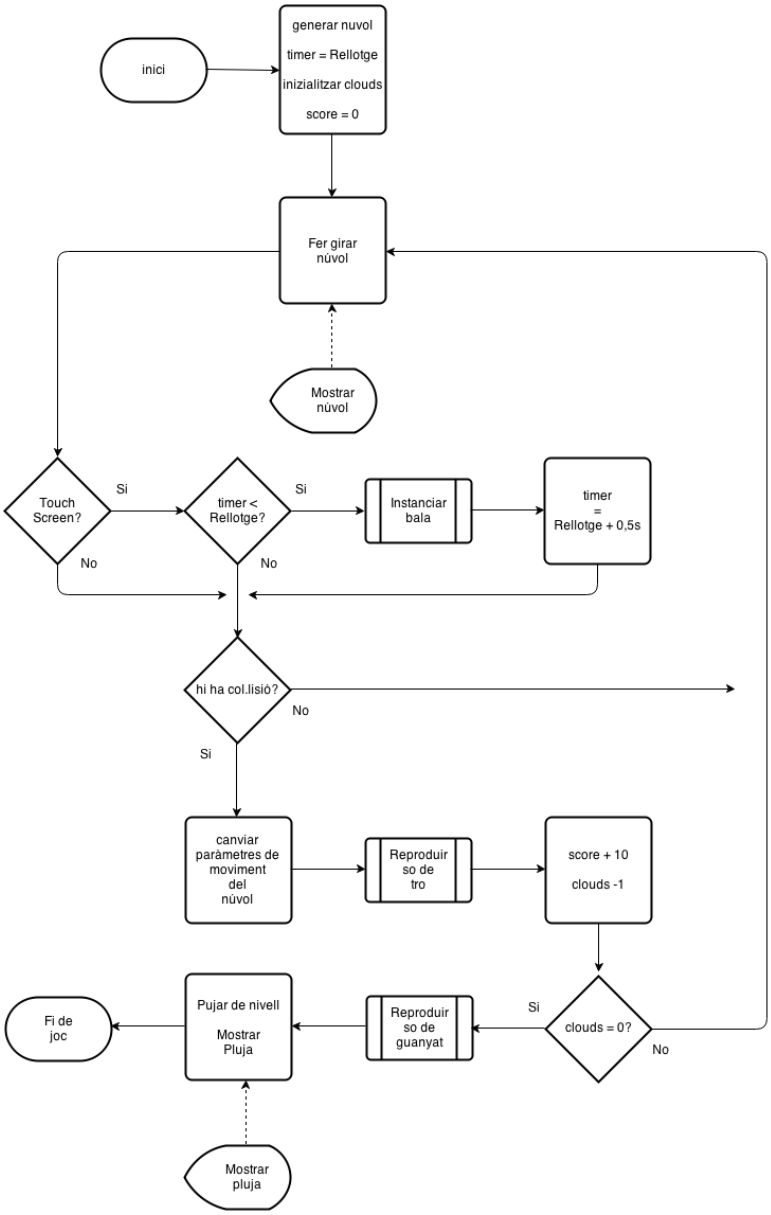


4.2.3. Diagrama de flux del joc de terra

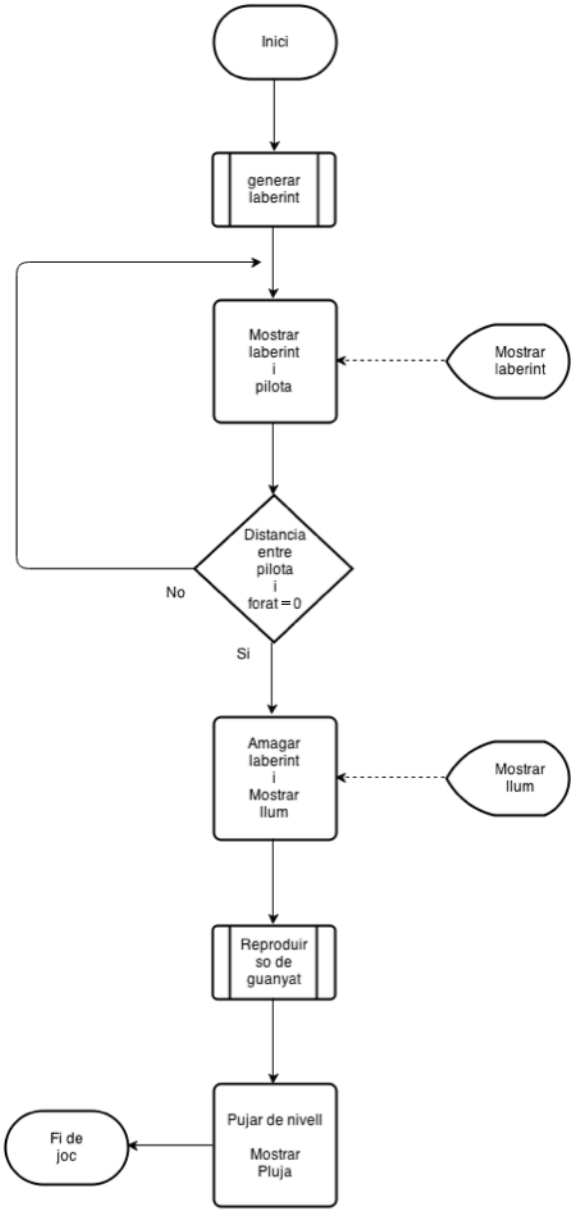



2
Delay
0.5 seg
contador <
Elements +1?
Si
No
Contador = 0
Estat =
mostrar?
No
Si
Contador + 1
Reproduir
so de color
Estat = resposta
Mostrar
color durant
1s
Mostrar
color durant
1s
4
3
Color escollit?
No
Si
Color [contador]
=
nul
No
Si
Score + 10
contador + 1
Color [contador]
=
Color escollit
color [contador]
correcte?
Si
No
contador <
Elements + 1?
Si
No
Estat = perdut
reproduir so
de perdut
Estat = guanyat
reproduir so
de guanyat
4
3
distancia entre
marcador i area
de color = 0?
vSo = on
vSo = on?
vSo = off
reproduir
so
de color
Mostrar Color
Mostrar Color
4

4.2.4. Diagrama de flux del joc d'aigua



4.2.5. Diagrama de flux del joc de llum



5. DISSENY DE LA INTERFICIE

5.1. Identitat corporativa

5.1.1. El logotip

El disseny del logotip intenta deixar clar l'argument de la aplicació. Es tracta de la imatge dibuixada d'una margarita. Aquest logotip serà la icona del joc un cop instal·lada la aplicació. L'usuari pot reconèixer l'aplicació immediatament mitjançant aquesta flor i ja es fa a la idea de que el joc tracta sobre plantes.

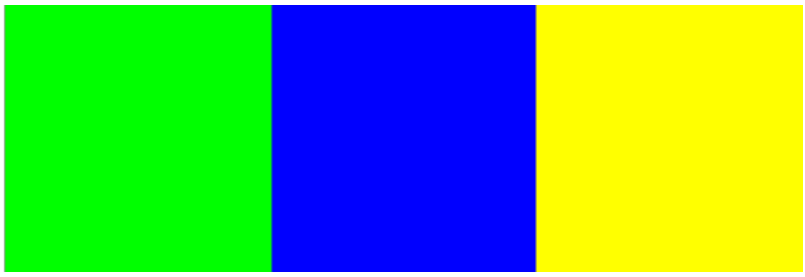
Figura 13 – Proves de logotip

5.1.2. Metàfora gràfica

Degut a que la aplicació està pensada per a gent de totes les edats. Però s'ha escollit una estètica de dibuixos on els colors base fan que sigui més atractiva per als usuaris de menys edat. Com l'argument del joc ens parla de les diferents parts que necessitem per al cultiu de plantes, l'escenari del joc es presenta com un petit hort. Aquest escenari es a on escollim un dels tres mini-jocs.

5.1.3. Colors corporatius

Tenim tres colors corporatius quan ens trobem amb el joc. Per un cantó tenim el verd i blau que van lligants amb els colors de la naturalesa. Per un altre cantó s'ha escollit el color groc per practicitat. Aquest color ens dona millor contrast respecte a les imatges preses per la càmera quan estem immersos en la RA.



hexadecimal: 00FF00 hexadecimal: 0000FF hexadecimal: FFFF00

Figura 14 – Colors corporatius

Verd

El verd s'ha escollit per a representar el camp del nostre hort a la pantalla de nivells. Pràcticament quasi tot el fons. També forma part dels dissenys de les plantes, tant el cactus com la margarita que ens trobarem a la realitat augmentada i com no cal dir a un dels quadrants del joc Simon.

Blau

El blau el podrem trobar tant al quadrant del joc Simon, com a quasi tots els textos que tenen interactivitat. Com en el cas dels botons start o quit game. Amb els botons de music y play quan es troben actius. També apareixen en molts textos informatius: tutorials o el nivell que ens trobem i quin tipus de mini-joc escollim.

Groc

El groc es pot trobar com el cor del logotip ja que es el centre de la margarita. Per altre banda també s'ha fet servir per als textos que hi ha a la RA ja que pot oferir-nos el millor contrast amb la realitat captada per la càmera. Un altre lloc on el trobem es al model 3D de la llum que en apareix un cop complert el mini-joc de la llum.

5.1.4. Tipografia

Per al títol o nom de l'aplicació s'ha fet servir la tipografia Kingthings Annex. Aquesta tipografia s'ha escollit ja que ens fa pensar en plantes.

Figura 15 – Tipografia Kingthings Annex

Només s'ha utilitzat al títol ja que el text és gran i com és un tipus de lletra el qual la seva lectura resulta complicada s'ha decidit no fer mes usos d'ella.

Per als botons on hi ha interacció s'ha triat la lletra thin cool. Aquesta lletra es fàcil de llegir i al ser un estilitzada permet ficar paraules llargues en espais petits.

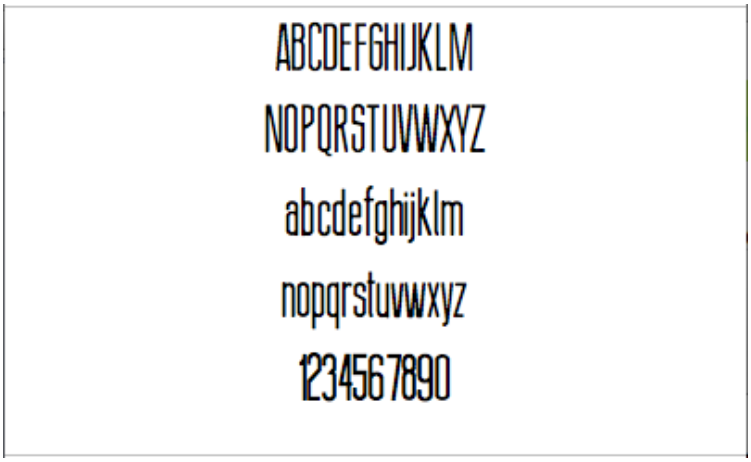


Figura 16 – Tipografia Thin Cool

Per als tota la resta de text: tutorials, missatges informatius, panell de punts, etc... on no hi ha interacció s'ha triat la tipografia source sans pro ja que és fàcil de llegir.

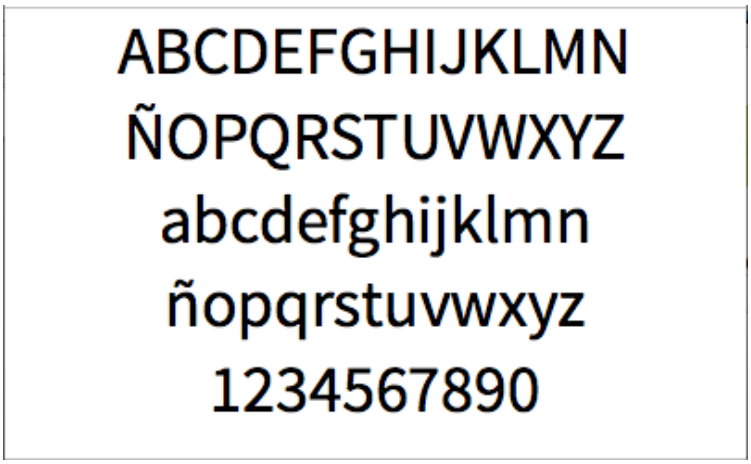


Figura 17 – Tipografia Source Sans Pro

6. HISTORIA I ENTORN

6.1. Història i narrativa

6.1.1. Història de fons

En el joc del projecte es vol fer que l'usuari entengui molt ràpid que es trobarà amb tres tipus de jocs diferenciats. La manera com s'ha escollit ha sigut agafant com a base el cultiu de les plantes.

L'aplicació es divideix en tres tipus de nivells: terra, aigua i llum. Com passa amb el cultiu de plantes necessiten una bona terra, llum i aigua per a fer-les créixer.

La terra ens ajuda a presentar el joc de memòria Simon on els diferents components de la terra es troben representats en 4 colors. L'aigua ens dona l'argument necessari per a entendre un joc de dispars. Disparant als núvols aconseguim que plogui. I finalment li donarem llum a la nostra planta. En el cas de la llum necessitem repara-la ja que la tenim espatllada d'un inici. Per fer-ho hem de solucionar el problema que té completant el laberint.

Els tres primers nivells de joc són tutorials. Aquest nivells no només ens serveixen per a explicar com juguem sinó que també s'aprofita per a mostrar el fil narratiu del joc.

6.2.1. Elements de la trama

Terra

El joc de la terra es troba representada amb un icona d'un test de plantes.

Figura 18 – icona de terra de Pots&Plants

Avanç de poder conrear rés a un test necessitem omplir-lo de terra. Però les nostres plantes no creixen a qualsevol tipus de terra. És necessari fer aquesta terra amb el correcte substrat, matèria orgànica, que contingui també minerals i vitamines per a que les nostres plantes creixin fortes. Es així que trobarem 4 colors al joc de Simon ja que identifiquem 4 components per a la terra.

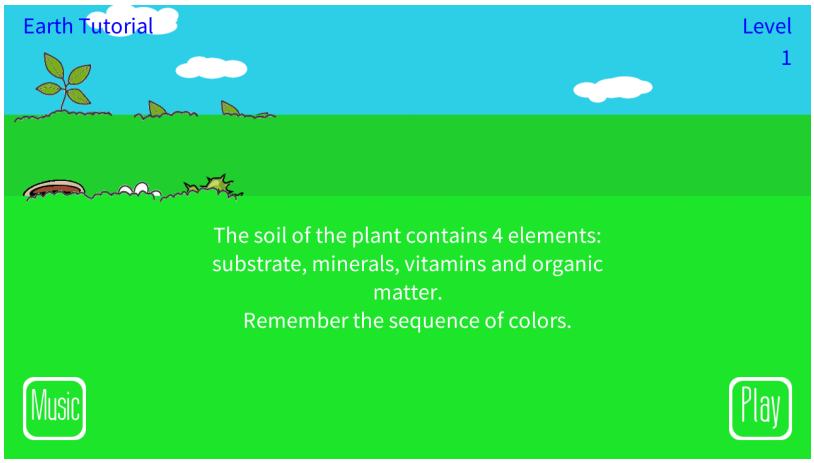


Figura 19 – Level 1 Pots&Plants

Avanç d'aconseguir completar el primer nivell del joc. A la pantalla de nivells ja se'ns expliquen aquests arguments de l'historia que anem assolint.

Figura 20 – Earth tutorial de Pots&Plants

Un cop iniciat el tutorial podrem llegir una mica més d'informació sobre la historia del joc. En aquest cas se'ns demanava que féssim la millor terra possible. Com? Aconseguint completant el joc.

Aigua

El joc d'aigua es troba representat amb un la icona d'una flor.

Figura 21 – icona d'aigua de Pots&Plants

Dintre del joc tenim les flors. Així s'explica al nivell 2 que les plantes necessiten per a realitzar la fotosíntesis.

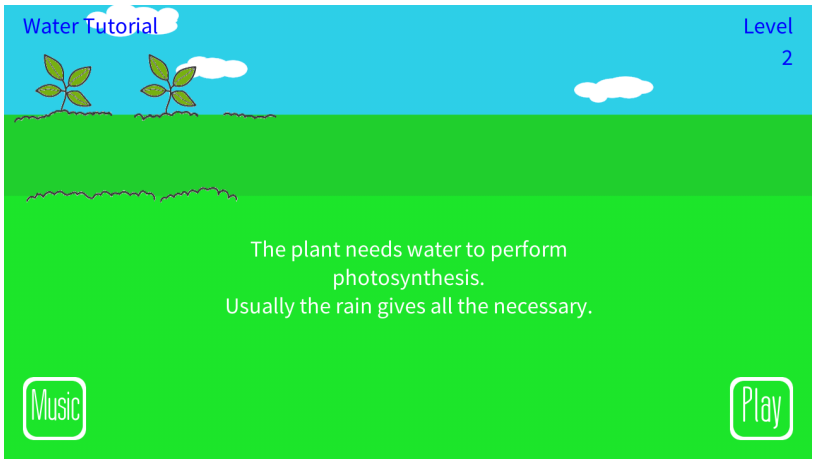


Figura 22 – Level 2 de Pots&Plants

El mateix passa amb el tutorial del segon nivell on trobarem més informació de l'argument. Aquí se'ns explica per que disparem als núvols en comptes de regar amb una mànega de jardí ja que sempre és millor l'aigua de la pluja que no pas la de l'aixeta. Tanmateix les plantes necessiten grans quantitats d'aigua per a créixer fortes.

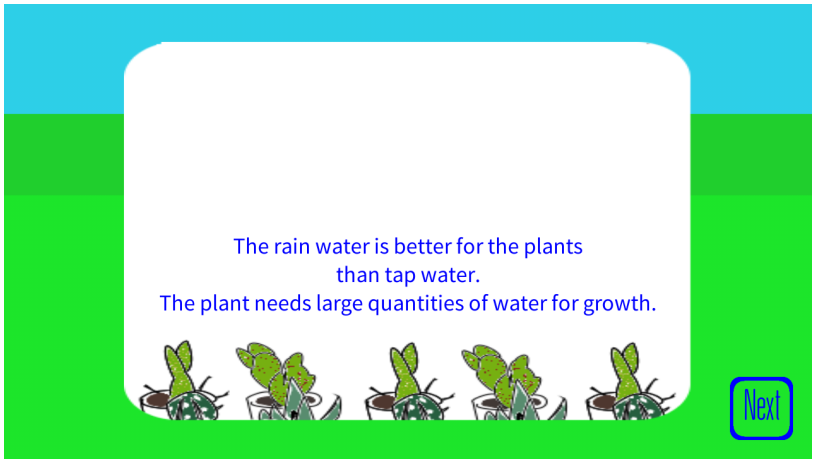


Figura 23 – Water tutorial de Pots&Plants

Llum

El joc de la llum es troba representat per la icona d'un cactus.

Figura 24 – icona de la llum de Pots&Plants

Els cactus com tothom hi pensa es troben al desert. I al desert hi ha poca aigua, la terra no te nutrients però hi ha molta quantitat de llum.

A la pantalla de nivells, quan ens trobem al nivell 3 de joc, s'explica que les plantes necessiten llum per a fer la fotosíntesis.

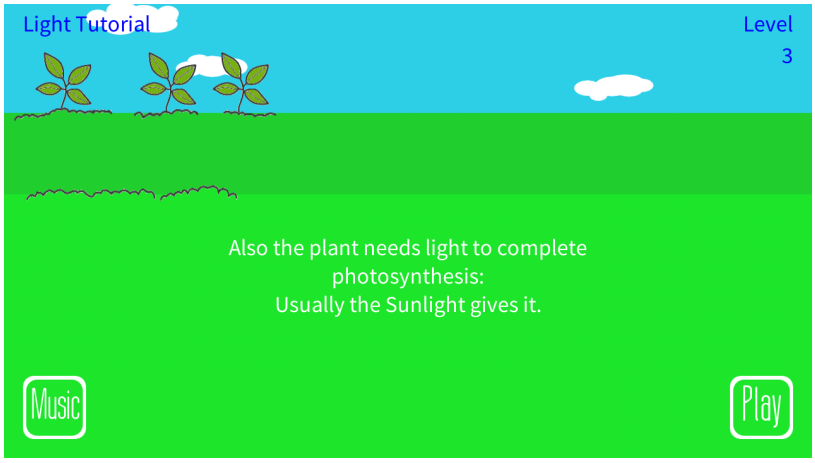


Figura 25 – Level 2 de Pots&Plants

Amb el tutorial de la llum, a l'iniciar la partida del nivell 3, sen's explica que hi ha llums artificials fetes per a estimular el creixement de les nostres plantes. Però la nostra es troba espatllada. I com s'ha dit a l'apartat 6.1.1 l'hem d'arreglar.

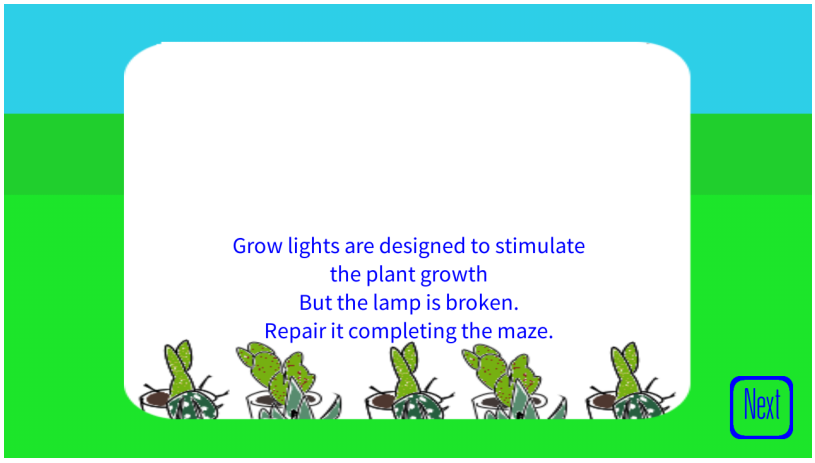


Figura 26 – Light tutorial de Pots&Plants

7. DISSENY DE PANTALLES I PERSONATGES

7.1. Disseny de pantalles

7.1.1. Pantalla de benvinguda

A la pantalla de benvinguda ens trobem amb el nom de l'aplicació al centre i dues opcions a escollir: start game o quit game. Aquesta pantalla es de fons blanc amb dibuixos d'unes plantes als voltants.

Figura 27 – Pantalla de benvinguda

7.1.2. Pantalla de nivells

A la pantalla de nivells en trobem un petit hort de 6 plantes i un test. Cada una de les plantes apareixen a mesura que augmentem el nostre nivell de 1 a 4. Durant els 3 primers nivells ens sortiran els 4 trebols que no son interactius. Al arribar al nivell 4 hem complert tots els tutorials així que ja podrem decidir pel nostre compte quin mini-joc escollim per jugar.

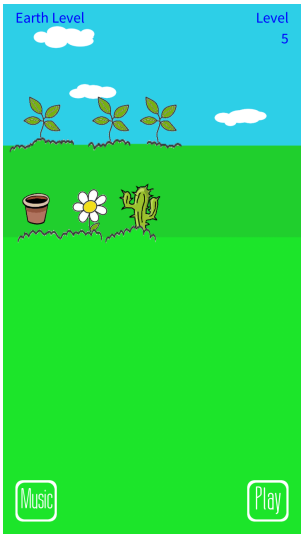


Figura 28 – Pantalla de nivells

A aquesta pantalla a la part superior tenim la informació de quin tipus de joc hem escollit i a quin nivell ens trobrem. Mentre que a la part inferior tenim els botons. A l'esquerra un botó per treure o deixar la cançó i a la dreta per jugar al mini-joc.

En el centre tindrem petits textos que ens expliquen l'argument del joc durant els tres primers nivells. A partir del 4 nivell es pot llegir un missatge que ens demana que seleccionem un dels 3 icones de les plantes per escollir el tipus de repte.

7.1.3. Pantalla de joc

Aquesta pantalla es el cor de la aplicació. Encara que dintre del joc hi ha 3 tipus de mini-jocs diferents, tot ells, comparteixen un mateix format de pantalla de joc.

A la pantalla de joc hi ha:

- 3 botons que el jugador pot identificar amb 3 icones.
- Un score de color groc on pot conèixer la puntuació .

- Un record de puntuació de color blanc. Encara que al mini-joc de la llum al tenir laberints completament diferents cada cop no s'ha establer un espai per al record de temps trigat.

Els tres icones tenen escollida la seva estètica segons la seva funció. Per exemple, una bombeta és el botó per posar o treure el flash de la càmera, una càmera per pendre una foto i un interrogant per activar l'ajuda.

Figura 29 – Pantalla de joc

Com els mòbils tenen una pantalla petita i a més a més ens trobem davant d'un escenari de RA, necessitem el màxim espai de pantalla per al joc, Per això tenim als cantons aquests botons i de la manera menys intrusiva la puntuació.

7.1.4. Pantalla d'ajuda

Un cop seleccionat el botó de l'ajuda del mini-joc podrem veure una pantalla amb un títol, el missatge d'ajuda i el texte que ens demana que toquem la pantalla per tornar al joc.

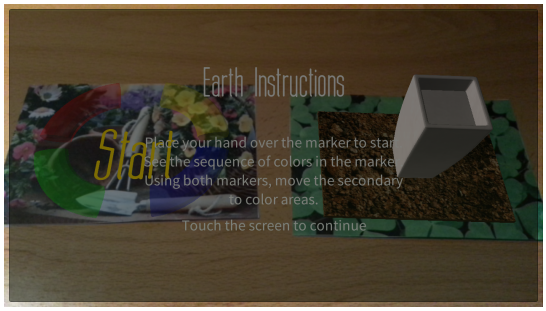


Figura 30 – Pantalla d'ajuda

Aquesta pantalla està feta dintre de la pantalla de Realitat Augmentada i amb una fons semitransparent per a que l'usuari no deixi de tenir la sensació que està interactuant amb la realitat.

7.2. Disseny de personatges

Tenim 2 entorns dintre de la aplicació. Un en 2D a la pantalla de nivells i dintre de la RA en 3D això fa que podem trobar els personatges del fil argumental en totes dues representacions.

Test

El test representa a la terra de les plantes, és qui ens permet identificar el mini-joc de Simon.

Test 2D

Test 3D

Figura 31 – Representacions de Test

Margarita

La flor de margarita no només és la icona o logotip de la aplicació, sinó que també és un personatge del joc. La flor es troba lligada al tema de l'aigual i com a personatge de joc té les dues representacions.

Margarita 2D

Margarita 3D

Figura 32 – Representacions de Margarita

Cactus

El cactus s'associa amb tema de la llum també té les dues representacions.

Cactus 2D

Cactus 3D

Figura 33 – Representacions de Cactus

7.2. Disseny de marcadors

Com s'ha comentat a l'apartat 2.3.2 s'han fest servir els Image targets com a marcadors. Per una altra banda tenim interactivitat dins dels propis marcadors. Al cas del mini-joc de terra hi ha botons virtuals al centre del imatge target que en permeten iniciar el joc o mirar la solució si hem fallat.

Aquí s'ha detectat dues condicións amb els virtual buttons:

- Han d'estar dins del marcador.
- Han d'estar situats en una zona de color. No funcionen si hi són a sobre d'una zona de color blanc.

Així que s'ha decidit per ficar imatges que omplen completament el marcador.

El joc requereix de l'ús de 2 marcadors per al correcte funcionament.

Figura 34 – marcador principal de Pots&Plants

El marcador primari és el que ens mostra els personatges dintre de l'entorn de RA. I on agafa més pes és als mini-jocs de terra i aigua. En el cas del mini-joc de llum només ens mostra el model del cactus.

L’altre marcador és el secundari. Es qui tindrà el model del Simon al joc de terra i amb el joc de llum el laberint i la llum.

Figura 35 – marcador secundari de Pots&Plants

8. CONCLUSIONS

8.1. Generales i/o especifiques

Realitzar un projecte d'aquestes dimensions es una feina complexa. Cal entendre que a l'inici del projecte no hi havia cap joc definit i tampoc es tenien els coneixements del funcionament de Vuforia.

Una de les part crítiques va ser establir els mini-jocs. D'un inici es volia fer un joc sencer en Realitat Augmentada però la manca d'inspiració va ser un impediment per acançar. Es va fer un estudi de les diferents Apps que hi ha de RA amb marcadors; es van revisar tot tipus de jocs... fins i tot el scrabble, malauradament sense resultat com a inspiració. Fins que amb l'ajuda del tutor es van establir el tipus de mini-jocs per engegar el procés de creació. Per tant cal dir que la creativitat és l'aspecte base d'aquest tipus d'aplicació.

Després d'observar diferents usuaris es va decidir tornar a revisar les aplicacions que s'havien vist durant el benchmarking. Es va detectar una diferencia i un problema amb l'aplicació: amb l'ús habitual dels mòbils es fa servir la pantalla tàctil com la eina de treball amb el dispositiu. Per tant la interactivitat ha d'estar a la pantalla i l'usuari no ha de tocar els marcadors per a res. Per tant el joc de terra s'ha de plantejar d'una nova manera encara que per temps queda fora del projecte.

8.2. Futures implementacions

El producte final està correctament fet per a un dispositiu Samsung s3 ja que l'usuari pot engegar l'aplicació i fer-la servir fins que es cansi d'ella sense que pateixi un problema inesperat: que no es pengi, no hi hagi sorolls fora de temps, els models 3d i els seus desplaçament es veuen quan i on han d'estar, etc... Degut a que no es disposa de tablets o altres aparells no s'ha pogut testejar si hi ha problemes de proporcions. Per tant com a punt de partida per a futures implementacions s'ha de revisar el funcionament amb aparells de pantalles d'altres dimensions.

Recordant el que s'ha pogut llegir a les conclusions: el joc de terra s'ha de repensar per a que l'usuari tingui una experiència de joc més intuïtiva.

I com a punt final creatiu es pot transformar l'aplicació en un joc de tipus Tamagochi, on hem de tenir cura de la nostra planta depenent de les seves necessitats.

8.3. Valoració del resultat en relació als objectius plantejats

Crear un joc de Realitat Augmentada.

Fer una aplicació de RA era el motiu del projecte, s'ha aconseguit fer una aplicació on l'usuari no només pot trobar un model 3D a sobre del marcador sinó que també pot interactuar amb els marcadors. Depenent de on i com es troba el marcador i ha sons, animacions, etc...

Aprendre com funciona Vuforia

Amb aquest punt ha quedat clar que no només cal conèixer la tecnologia sinó que també s'ha de tenir en compte la seva fi. La implementació dintre de l'aplicació ha fer que sigui intuïtiu per a l'usuari tal com no succeeix amb el joc de terra.

Millorar amb mètodes de programació amb llenguatges orientats a objectes. En aquest cas C# on es va crear un menú delegate, com es fa amb OSX per a Iphone. O per passar dades d'un escenari a un altre s'ha creat un singleton, un mètode aprés també a programació de IOS. Es van assolir aquests reptes i es va aconseguir fer un treball coherent i funcional.

APÈNDIX

A.1. Apèndix Jocs.

4.1.1. Recettear an ítem Shop's Tale

Figura 36 – Recettear

Recettear: An Item Shop's Tale és un joc indie japonés. Està desenvolupat per EasyGameStation i combina l'estil RPG d'exploració amb joc de vendes. El joc va ser llançat i distribuït al Comiket 73 del desembre de 2007. La seva versió en anglès va aparèixer al setembre del 2010 guanyant una gran popularitat internacional.

L'argument del joc ens fa agafar el rol de Recette Lemongrass, filla d'un heroi que constantment es troba de viatge per salvar el món. Ara ve la diferencia d'aquest joc a altres: el nostre oponent és un banc a qui li hem de pagar el deute que ens ha deixat el pare de Recette.

Tenim un calendari de pagaments setmanal, temps de joc, amb unes quantitats específiques i cada cop més grans. Donat el cas que no podem pagar el deute, el banc es farà càrrec de la nostre casa i perdem el joc.

Com es tracta d'un joc de compra venda tenim un nivell de venedor. Aquest nivell ens dona més opcions de joc a mesura que l'anem incrementant.

Per aconseguir un producte tenim 3 opcions: comprar-lo al mercat o gremis i revendre'l; contractar un mercenari que explori les masmorres del joc recollint ítems per a nosaltres; o fabricar objectes millorats amb els articles que hem obtingut a les masmorres o comprat al mercat.

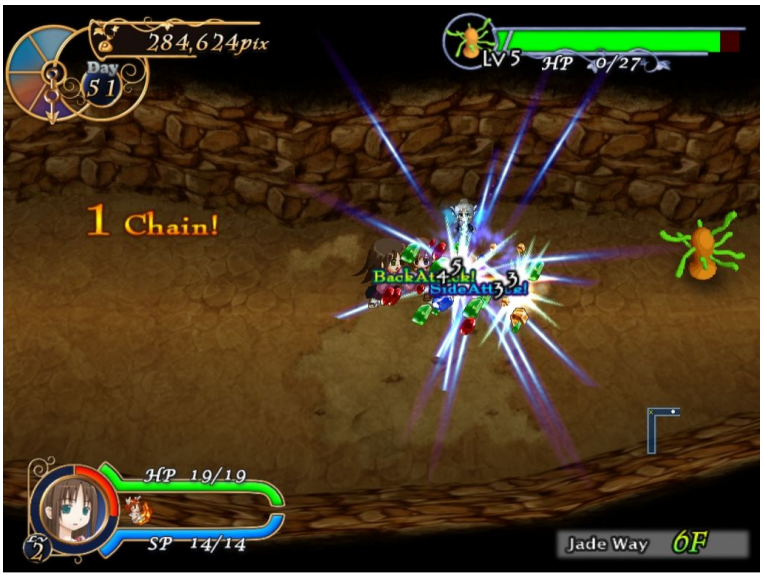


Figura 37 – masmorra de Recettear.

Cada matí ens trobem que els preus dels productes poden variar. De fet ens donen 3 missatges. Els dos primers ens diuen els productes que durant tot dia tindran augment de preu o patiran una davallada. I l'últim missatge ens diu quin es el producte de moda entre el públic.

Figura 38 – missatges de Recettear.

No oblidem que parlem d'un JRPG i per tant durant el joc van pareixent personatges nous que ens expliquen una historia per cadascun d'ells amb nous objectius.

Punts forts:

Moltes possibilitats de joc: Podem pensar que el "target" de jugador és indefinit. I moltes hores per aprendre tàctiques i tècniques per a assolir el deute i completar els mapes de les "dungeons".

Petits reptes: cada dia a la botiga ens poden fer comandes de productes que hem d'aconseguir tenir a curt termini. Per exemple: dues bufandes de llana que vindran a recollir en 2 dies.

Figura 39 – comandes de Recettear.

Reptes personals: Explorar l'historia de cada personatge implica fer viatges per les masmorres del món però no és obligatori ja que amb la revenda de productes podem assolir el deute setmanal. També cal dir que per a poder fabricar objectes hem de viatjar a les masmorres ja que tota fabricació requereix d'articles que només es troben a masmorres.

Punts febles:

No te un fil argumental fixe. Al tractar-se d'un joc de compravenda i el banc cada setmana ens augmenta el deute. Podríem dir que es un joc infinit sempre que aconseguim acomplir el deute. Per tant la partida mai s'acaba i s'esborra.

Aportació al projecte:

Desenvolupaments de petits reptes separats el jugador pot experimentar cada nivell sense vincles o requisits per accedir als diferents objectius del joc.

No hi ha límit de nivells al joc. Cada cop que guanyem augmentem el nombre d'encerts o es canvia el mapa per a que pugui fer un nou repte.

4.1.2. Carcassone

Condicions inicials:

Cada jugador col·locarà un dels seguidors a la casella 0 del marcador de puntuació.

Figura 40 – marcador de 3 jugadors.

Partida:

Els torns de la partida estan dividits en 3 fases: col·locació de parcel.la, col·locació de seguidor i puntuació.

Per poder col·locar una parcel.la, com a mínim un dels seus cantons ha de ser adjacent a una de les peces que hi ha a la taula.

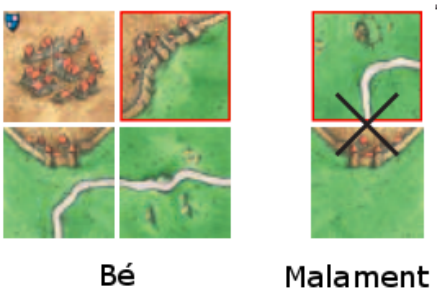


Figura 41 – Col·locació parcel.la

L'altre condició és que tots els segments de camp, ciutat o camí s'han de continuar o tancar amb segments del mateix tipus.

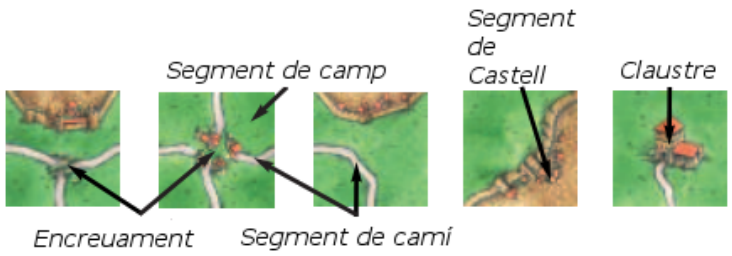


Figura 42 – Segments de parcel·les

A la segona part del torn anem a posar el meeple. Aquest ninot es pot deixar a un segment (ciutat, camí o camp) o a un claustre. A un segment de camp es col·loca tombat. Si el col·loquem al camí serà un lladre, al camp un granger, al ciutat un cavaller i al claustre un monjo.

Només hi ha una condició. No es pot col·locar un seguidor a un segment on hi ha un meeple.

Figura 43 – Seguidors al taulell.

A la tercera part del torn calculem els punt obtinguts si hi ha. Per a que hi hagi punts hem de tancar un segment a la primer part de torn. I hem de tenir un meeple al segment tancat a l'hora de puntuar.

Quan aconseguim punts per un camí, claustre o ciutat recuperem els seguidors i els podem continuar fent servir.

Els grangers només puntuen al recompte final per tant no es poden recuperar.

La partida acaba quan es col·loqui l'ultima peça de territori.

Punts per camí completat. Un camí es considera completat quan als extrems del camí trobem un encreuament, un ciutat o un claustre. S'anoten tants punts com caselles te el camí.

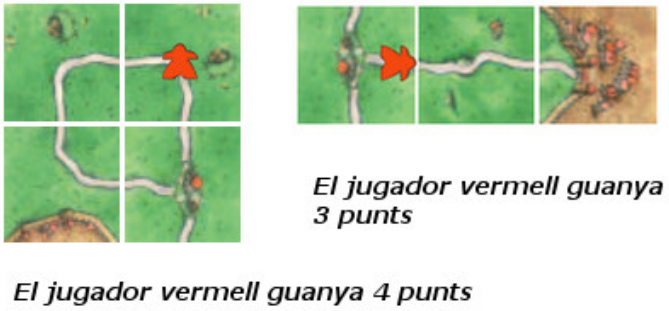


Figura 44 – Punts per camí.

Punts per ciutat completat. Una ciutat es considera complerta quan la muralla rodeja completament la ciutat. S'anoten 2 punts per casella que tingui la ciutat més 2 punts per escut que hi hagi dins de la ciutat.

Si la ciutat només és de 2 caselles. S'anota 1 punt per casella i escut que tingui la ciutat.

Figura 45 – Punts per ciutat.

Si a una ciutat, camí o camp hi ha més d'un meeple. El jugador que tingui més nombre de seguidors al territori guanya els punts. Si hi ha empat amb el nombre mes alt de seguidors, tots els qui empatin es porten els punts.

Punts per claustre. Un claustre es considera completat quan la peça del claustre està completament rodejada per peces. D'aquesta manera el jugador que tingui el monjo al claustre s'emporta 9 punts.

Figura 46 – Claustre complert.

Casella central rodejada del tot.

Punts per granja. Recordem que els punts dels grangers només es compten al final de la partida. Cada granja dona 3 punts per ciutat que abasteix. Només es compten les ciutats complertes que estiguin tocant el territori del granger.

Figura 47 – Punts per granges

El territori amb més meeples d'un jugador dona els punts a aquell jugador. En cas d'empat en el nombre màxim tot els jugadors que empatin es porten els punts.

Recompte de punts final. Primer es compten els territoris incomplerts i després es sumen els punts per granges.

La puntuació per territoris incomplerts es pot dir que és punt per casella. Es a dir, amb els claustres es suma 1 punt per la casella de claustre i 1 punt més per cada casella que tingui al voltant; per camins 1 punt per cada casella que tingui el camí; en el cas de les ciutats comptarem 1 punt per casella i escut.

Punts forts del joc:

Senzillesa : Les regles son poques i simples. Fa que sigui fàcil d'aprendre. Per exemple, a un territori ocupat per un meeple no es pot col·locar un altre.

Agilitat: El fet de que tingui pocs tipus de fitxes i condicions per a la seva col·locació fa que sigui un joc ràpid de jugar i per tant tingui enganxat al jugador pensant la seva estratègia.

Punts febles del joc:

Es un joc per jugar una estona curta degut a la seva agilitat i senzillesa. Pensar en perdre tota la tarda jugant al Carcassonne acaba sent avorrit.

Aportació al projecte:

Només cal tenir 3 tipus de condicions per col·locar una fitxa. Fa que sigui fàcil d'aprendre i àgil per a la decisió.

4.1.3. Archer - Nintendo DS: jocs RA.

La consola Nintendo 3DS inclou una aplicació i 6 marcadors per a jugar amb jocs RA. Aprofita les dues càmeres que porta la videoconsola junt amb les cartes per iniciar els jocs.

Figura 48 – Nintendo DS

El primer joc que ofereix aquest sistema es diu Archer i com en nom indica hem de disparar fletxes amb els botons de la consola a unes dianes. Les dianes es troben al lloc on hem col·locat el marcador.

Figura 49 – Nintendo Cards

El primer joc que es pot trobar es diu Archer. Amb aquest joc la funció de l'usuari es disparar a unes dianes que es troben a on s'ha col.locat el marcador.

Figura 50 – Archer

Un cop encertades totes les dianes surt el monstre final Un drac. El drac es ofensiu i ataca. Per a poder evitar els seus atacs l'usuari s'ha de moure.

Figura 51 – Drac Archer

Punts forts: Joc molt simple. Intuïtiu i senzill. L'usuari sap en tot moment que ha de fer seguint les instruccions que apareixen a la pantalla.

Primer contacte amb la realitat augmentada per part de l'usuari. Aquest punt afegeix valor a la marca de la consola que pot atraure consumidors.

Nivells de dificultat. Si l'usuari no es mou pot encertar totes les dianes i patir els atacs del drac. Si l'usuari vol ser més eficient por moure's per l'entorn que l'envolta.

Punt feble: joc molt curt i repetitiu. En menys de 3 minuts l'usuari ha complert l'objectiu final.

Aportació al projecte: Ja que segurament serà un primer contacte amb la Realitat Augmentada s'ha de donar a l'usuari tot el temps que necessiti per complir la missió sense establir límits de temps o dispars fallats.

4.1.4. Fishing - Nintendo DS: jocs RA.

Es tracta d'un joc de pesca on el marker transforma l'entorn en una piscina mentre que a la pantalla ens apareix una canya de pescar.

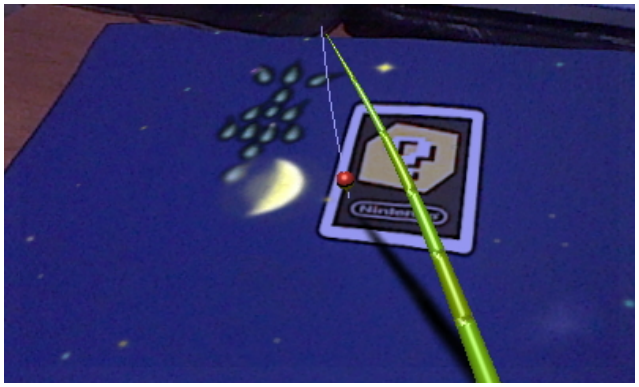


Figura 52 – Fishing

Tenim limitat el temps de joc a dos minuts un cop acabat el temps apareix el drac del joc Archer i ens ataca. L'objectiu del joc es col·leccionar el màxim nombre de peixos, es poden trobar fins a 31 tipus de peixos diferents.

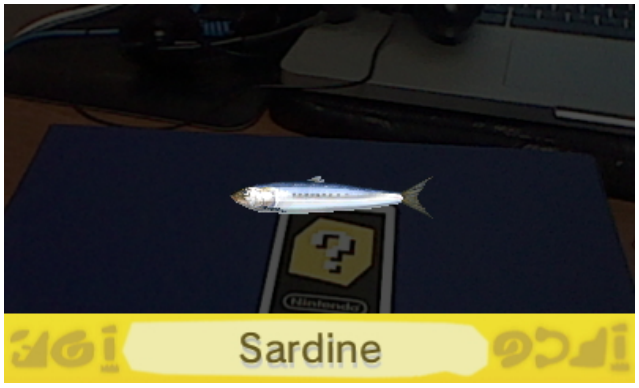


Figura 53 – Assoliment a Fishing

Poden succeir diferents esdeveniments depenent d'on tenim el marcador. Per exemple si la taula es d'un color vermell ens surten uns peixos diferents a si la taula es de color verd. També a l'iniciar el joc pot ser que caigui un raig i aleshores els peixos són peixos esquelets.

4.1.5. Billiard - Nintendo DS: jocs RA.

Aquest joc segueix una formula molt semblant a la d'un joc de dispars. Un cop carregat el joc, apareix a la pantalla un escenari amb muntanyes, rius i altres tipus d'obstacles. Com a disparador tenim un tac de billar que colpeja a una bola metàl·lica. L'escenari

pot estar en moviment on la inclinació del terreny varia augmentant el nivell de dificultat i destresa.

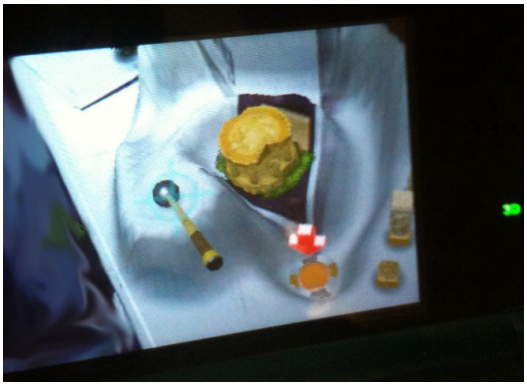


Figura 54 – Billiards

En aquest escenari l'objectiu del jugador es aconseguir arribar al forat amb la bola, però com si es tractés d'un joc de golf, on s'ha de fer amb el mínim nombre de cops. Així que l'objectiu real és fer el mínim de dispars.

Un cop complert l'objectiu ens torna a aparèixer el drac de Archer amb el que tindrem un combat. Com és evident l'arma es el tac i la munició les boles de billar.

Figura 55 – Drac Billiards

A.2. Apèndix links.

Pan European Game Information

http://es.wikipedia.org/wiki/Pan_European_Game_Information

Realitat augmentada

http://es.wikipedia.org/wiki/Realidad_aumentada
http://en.wikipedia.org/wiki/Augmented_reality
http://www.qualcomm.com/solutions/augmented-reality
https://developer.vuforia.com/resources/sdk/unity
http://www.bihartech.com/ejemplos-realidad-aumentada/
http://www.marque.es/blog/?p=166

Jocs RA de NINTENDO DS

https://www.nintendo.es/Nintendo-3DS/Software-instantaneo/Juegos-RA-la-realidad-aumentada/Juegos-RA-la-realidad-aumentada-115169.html

Archer GamePlay https://www.youtube.com/watch?v=lKS0caNPR4o

Fishing GamePlay https://www.youtube.com/watch?v=C33_mSf8pNY

Billiards GamePlay https://www.youtube.com/watch?v=J01Uc7j-dRg

Jocs RA de PS VITA

http://www.godisageek.com/2012/03/playstation-vita-ar-games-review
http://www.youtube.com/watch?v=OElik_UM01w
http://www.youtube.com/watch?v=kEMDgvfFUcI

Joc Drakerz

http://www.drakerz.com/
https://www.youtube.com/watch?v=yEaR116swnQ

ART http://es.gizmodo.com/ping-pong-y-realidad-aumentada-o-como-crear-un-juego-d-755123990

Projectes

Oggboard https://www.kickstarter.com/projects/samandbrock/the-oggboard?ref=category

React Table http://www.youtube.com/watch?v=QflrIK-m4Ts#t=144

Crushing Darkness: an augmented reality trading card game

https://www.kickstarter.com/projects/979861484/crushing-darkness-an-augmented-reality-trading-car?ref=category

Deep Green & Augmented Reality Pool

https://www.youtube.com/watch?v=AENJxqR0g48

Joc Simon http://ca.wikipedia.org/wiki/Simon_(joc)

Icones

http://findicons.com/icon/73368/light_bulb_on

http://findicons.com/icon/73413/light_bulb_off

http://findicons.com/icon/52398/scanners_and_cameras?id=52398

Cançons i sons

http://content.gpwiki.org/index.php/Game_Content_Resources#Music

http://incompetech.com/music/royalty-free/index.html?feels%5B%5D=Calming

Simon

http://www.webdesign.org/img_articles/11082/simonSound1.zip

http://www.webdesign.org/img_articles/11082/simonSound2.zip

http://www.webdesign.org/img_articles/11082/simonSound3.zip

http://www.webdesign.org/img_articles/11082/simonSound4.zip

Win/Loose

http://soundbible.com/1830-Sad-Trombone.html

http://soundbible.com/1003-Ta-Da.html

http://www.mp3louder.com/

A.3. Apèndix Codi.

Autorespawn.CS

```
using UnityEngine;
using System.Collections;

public class AutoRespawn : MonoBehaviour {

    public bool cloudDown = false;
    private float timeWait = 0;

    void Update ()
    {
        if (cloudDown && GameObject.Find("Cube_000").renderer.enabled)
        {
                if (timeWait == 0)
                    timeWait = Time.time + Random.Range(1,3);
                else
                {
                        if (timeWait < Time.time)
                        {
                                timeWait = 0;
                                gameObject.renderer.enabled = true;
                                gameObject.collider.enabled = true;
                               cloudDown = false;
                        }
                }
        }
    }
}
```

BallMoves.cs

```
using UnityEngine;
using System.Collections;

public class BallMoves : MonoBehaviour {

    public Transform hole;
    public GUIText myTime;
    public GameObject myLamp;
    public AudioClip winner;
    private bool winning = false;
    private float timer = 0;

    void LoadLevelsScene()
    {
        Application.LoadLevel ("Levels");
    }

    void Update () {
        if (gameObject.GetComponent<MeshRenderer> ().enabled && gameObject.GetComponent<Rigidbo
dy> ().isKinematic)
                gameObject.GetComponent<Rigidbody> ().isKinematic = false;
        else
            if (!gameObject.GetComponent<MeshRenderer> ().enabled)
                    gameObject.GetComponent<Rigidbody>().isKinematic = true;

        float dist = Vector3.Distance (hole.position, gameObject.transform.position);

        if (dist < 25 && !winning)
        {
            MySingleton.Instance.changeLevel = true;
            audio.PlayOneShot(winner);
            winning = true;
            Invoke("LoadLevelsScene",5);
            GameObject.Find("Maze").SetActive(false);
            gameObject.renderer.enabled = false;
            myLamp.SetActive(true);
        }
```

```
if (gameObject.renderer.enabled)
        {
            timer += Time.deltaTime ;
            int minutes = (int)timer / 60;
            int seconds = (int)timer % 60;
            int fraction = (int) (timer*100) % 100;
            myTime.text = (minutes.ToString() + ":" +seconds.ToString()+"."+fraction.ToString()) ;
        }
    }
}
```

ButtonColors.CS

```
using UnityEngine;
using System.Collections;

public class ButtonColors : MonoBehaviour {

    // Update is called once per frame
    void Update () {

        SymonThread myThread = GameObject.Find ("Seed").GetComponent<SymonThread> ();
        Transform imageMarker = GameObject.Find ("Seed").transform;
        float dist = Vector3.Distance (transform.position, imageMarker.position);

        if (dist < 450 && GameObject.Find ("UNDER").renderer.enabled && GameObject.Find("TextStart").r
enderer.enabled)
        {
            myThread.release = true;
            switch (gameObject.name)
            {
                case "ButtonRed":
                    myThread.buttonPressed = 0;
                    break;
                case "ButtonGreen":
                    myThread.buttonPressed = 1;
                    break;
                case "ButtonBlue":
                    myThread.buttonPressed = 2;
                    break;
                case "ButtonYellow":
                    myThread.buttonPressed = 3;
                    break;
            }
        }

        if (dist > 450 && myThread.release)
        {
            string myName = "";
            switch (myThread.buttonPressed)
            {
                case 0 :
                    myName = "ButtonRed";
                    break;
                case 1 :
                    myName = "ButtonGreen";
```

```
                    break;
                case 2 :
                    myName = "ButtonBlue";
                    break;
                case 3 :
                    myName = "ButtonYellow";
                    break;
            }
            if (gameObject.name == myName)
            {
                myThread.release = false;
            }

        }

        if (GameObject.Find ("UNDER").renderer.enabled)
            gameObject.GetComponent<MeshRenderer> ().enabled = true;
    }
}
```

ButtonLevel.CS

```
using UnityEngine;
using System.Collections;

public class ButtonLevel : MonoBehaviour {

    public float x;
    public float yRatio;
    public string levelToGo;

    void OnGUI()
    {
        //GUI.color = Color.clear;
        if (GUI.Button (new Rect (x, Screen.height*(1-yRatio)-120, 100, 120), ""))
        {
            Application.LoadLevel (levelToGo);
        }
    }
}
```

```
using UnityEngine;
using System.Collections;

public class ButtonStart : MonoBehaviour , IVirtualButtonEventHandler{

    // Use this for initialization
    public GameObject simon;
    private int loop;

    void Start () {

        VirtualButtonBehaviour[] btn = transform.GetComponentsInChildren<VirtualButtonBehaviour> ();
        foreach (VirtualButtonBehaviour bbs in btn)
        {
            bbs.RegisterEventHandler(this); //Vuforia documentation web --> Register an event handler
        }

    }

    //Problemes Si els botons no tenen color base poden fallar. Tots a parts on el dibuix te color.

    public void OnButtonPressed (VirtualButtonAbstractBehaviour vb)
    {
        if (GameObject.Find("Seed").GetComponent<SymonThread>().gameStat == SymonThread.State.Of
f)
        {
            GameObject.Find("Seed").GetComponent<SymonThread>().gameStat = SymonThread.State.Cre
ate;
        }

        if (GameObject.Find ("Seed").GetComponent<SymonThread> ().gameStat == SymonThread.State.L
oose)
        {
            GameObject.Find ("Seed").GetComponent<SymonThread> ().reDoSequenze();
        }
    }

    public void OnButtonReleased (VirtualButtonAbstractBehaviour vb)
    {
        gameObject.SetActive (false);
    }

}
```

DelegateMenu.CS

```
private delegate void MenuDelegate ();
private MenuDelegate menuFunction;

private float screenHeight;
private float screenWidth;
private float buttonHeight;
private float buttonWidth;
public Texture2D background;
private bool button1Clicked = false;
private bool button2Clicked = false;

void Start()
{

    //start in landscape mode
    Screen.orientation = ScreenOrientation.Portrait;

    //variables of area dimensions
    screenHeight = Screen.height;
    screenWidth = Screen.width;

    //al tornar desde un level landscape fa que surti malament.
    if (screenHeight < screenWidth)
    {
        screenHeight = Screen.width;
        screenWidth = Screen.height;
    }

    buttonHeight = screenHeight * 0.2f;
    buttonWidth = screenHeight * 0.8f;

    menuFunction = mainMenu;

}

void OnGUI ()
{

    menuFunction();
    if (Input.GetKeyDown(KeyCode.Escape)) { Application.Quit(); }
}
```

```
    void mainMenu()
    {
        //background
        GUI.DrawTexture(new Rect(0, 0, screenWidth , screenHeight),background);

        //setting fonts
        GUI.skin.label.font = (Font)Resources.Load("Fonts/Titles", typeof(Font));
        GUI.skin.button.font = (Font)Resources.Load("Fonts/Thin Cool", typeof(Font));
        GUI.skin.label.fontSize = (int) screenWidth/7;
        GUI.skin.button.fontSize = (int) screenHeight/10;

         //setting Title
        GUI.color = new Color(0F, .5F, 0F);
        GUI.skin.label.alignment = TextAnchor.MiddleCenter;
        GUI.Label (new Rect (0, screenHeight/2, screenWidth, buttonHeight),"Pots & Plants");


        //set button1 color
        GUI.backgroundColor = Color.clear;
        GUI.color = Color.blue;
        if(button1Clicked)
            GUI.color = Color.green;

        //button1 pressed
        if (GUI.Button(new Rect ((screenWidth - buttonWidth) * 0.5f, screenHeight/2 -
 buttonHeight, buttonWidth, buttonHeight),"Start Game"))
        {
            Application.LoadLevel(1);
            button1Clicked= true;
        }

        GUI.color = Color.blue;
        if(button2Clicked)
            GUI.color = Color.green;

        //button2 pressed
        if (GUI.Button(new Rect ((screenWidth -
 buttonWidth) * 0.5f, screenHeight/2 + buttonHeight, buttonWidth, buttonHeight),"Quit Game"))
        {
            button2Clicked= true;
            Application.Quit();
        }
    }

}
```

DestroyAfterTime.CS

```
using UnityEngine;
using System.Collections;

public class DestroyAfterTime : MonoBehaviour {

    // Use this for initialization
    void Start () {
        Destroy (gameObject, 5f);
    }
}
```

FireShotScript.CS

```
using UnityEngine;
using System.Collections;

public class FireShootScript : MonoBehaviour {

    private int axisZ = 200;

    void Update () {
        if (MySingleton.Instance.pause)
            axisZ = 0;
        else
            axisZ = 200;
        transform.Translate (0,0,axisZ);
    }

    void OnTriggerEnter (Collider other)
    {
        if (other.name == "PrefabCloud")
        {
                GameObject.Find ("shootSpawn").GetComponent<ShotScript>().scored = true;
                GameObject.Find ("PrefabCloud").GetComponent<AutoRespawn>().cloudDown = true;
                other.renderer.enabled = false;
                other.collider.enabled = false;
        }
    }
}
```

LoadLevel.CS

```
using UnityEngine;
using System.Collections;

public class LoadLevel : MonoBehaviour {

    public int level;
    private float timeLapse;
    private float grewUp;

    // Use this for initialization

    void Start ()
    {
        if (MySingleton.Instance.myMusic)
                MySingleton.Instance.MusicOn ();

        Screen.orientation = ScreenOrientation.AutoRotation;
        level = PlayerPrefs.GetInt ("Level");
        GameObject.Find ("TextLevel").guiText.text = level.ToString();
        grewUp = (float) 130 / Screen.height;
        int maxLevel = 6;
        if (level > 3)
            maxLevel = 3;

        for (int i = 1; i <= maxLevel ; i++)
        {
            GameObject go = GameObject.Find("Plant" + i);
            if (level < i)
            {
                go.transform.Translate(0,-grewUp, 0);
            }
        }

        timeLapse = 0;

        if (level == 0)
            MySingleton.Instance.changeLevel = true;

    }

    void Update()
    {
```

```
        if (MySingleton.Instance.changeLevel)
        {
                level ++;
                PlayerPrefs.SetInt ("Level", level);
                GameObject.Find ("TextLevel").guiText.text = level.ToString ();
                timeLapse = Time.time + 1f;
                MySingleton.Instance.changeLevel = false;
        }

        if (Time.time < timeLapse && level < 5)
        {
                NewPlant (level);

                if (level == 4)
                {
                   NewPlant (5);
                   NewPlant (6);
                   GameObject.Find ("TextTypeLevel").guiText.text = "Earth Level";
                   GameObject.Find ("TextExplanation").guiText.text = "Click on icons to select earth, water or li
ght";
                }

        }


        if (Input.GetKeyDown(KeyCode.Escape)) { Application.LoadLevel(0); }
    }


    void NewPlant(int i)
    {
       GameObject go = GameObject.Find ("Plant" + i);
       go.transform.Translate (0, Time.deltaTime * grewUp, 0);
    }
}
```

MazeGenerate.CS

```
using UnityEngine;
using System.Collections;

public class Mazegenerate : MonoBehaviour {


    public int mazeHeight = 11;
    public int mazeWidth = 11;
    private int [,] myMaze;

    public GameObject WallMaze;

    private static System.Random _random = new System.Random();
    // Use this for initialization
    void Start () {

        myMaze = GenerateMaze (mazeHeight,mazeWidth);

        for (int i = 0; i < mazeHeight; i++)
        {
            for (int j = 0; j<mazeWidth; j++)
            {
                    if (myMaze [i, j] == 1)
                    {
                        Vector3 pos = new Vector3 ((i * 50)-300, 25, (j * 50)-175);
                        GameObject wall = Instantiate (WallMaze,pos,gameObject.transform.rotation) as Game
Object;
                        wall.transform.parent = gameObject.transform;
                        wall.AddComponent<BoxCollider>();
                    }
            }
        }
    }


    private int [,] GenerateMaze(int height, int width)
    {
        myMaze = new int[height,width];
        for (int i = 0; i < height; i++)
            for (int j = 0; j < width; j++)
                    myMaze [i, j] = 1;

        System.Random rand = new System.Random ();
```

```
    int h = rand.Next (height);
    while (h%2 ==0)
        h = rand.Next (height);
    int w = (int) rand.Next (width);
    while (w%2 ==0)
        w = rand.Next (width);
    myMaze [h, w] = 0;

    MazeDigger (myMaze, h, w);

    return myMaze;

}

private void MazeDigger (int [,] maze, int h, int w)
{
    int[] directions = new int[] {1,2,3,4};
    Shuffle (directions);
    //    Debug.Log ("Direcciones" + directions [0] + directions [1] + directions [2] + directions [3]);
    for (int i=0; i<directions.Length; i++)
    {
        switch(directions[i])
        {
        case 1:
            if (h - 2 <= 0)
                continue;
            if(maze[h-2,w] !=0)
            {
                maze[h-2,w]=0;
                maze[h-1,w]=0;
                MazeDigger(maze, h-2,w);
            }
            break;
        case 2:
            if (w + 2 >= mazeWidth - 1)
                continue;
            if(maze[h,w+2] !=0)
            {
                maze[h,w+2]=0;
                maze[h,w+1]=0;
                MazeDigger(maze, h,w+2);
            }
            break;
        case 3:
```

```
            if (h + 2 >= mazeHeight - 1)
                continue;
            if(maze[h+2,w] !=0)
            {
                maze[h+2,w]=0;
                maze[h+1,w]=0;
                MazeDigger(maze, h+2,w);
            }
            break;
        case 4:
            if (w - 2 <= 0)
                continue;
            if(maze[h,w-2] !=0)
            {
                maze[h,w-2]=0;
                maze[h,w-1]=0;
                MazeDigger(maze, h,w-2);
            }
            break;
        }
    }

}

public static void Shuffle<T>(T[] array)
{
    var random = _random;
    for (int i = array.Length; i>1; i--)
    {
        int j = random.Next(i);
        T tmp = array[j];
        array[j] = array [i-1];
        array[i-1]= tmp;
    }
}
}
```

MusicScript.CS

```
using UnityEngine;
using System.Collections;

public class MusicScript : MonoBehaviour {

    public GUIStyle myStyle = new GUIStyle();

    void OnGUI()
    {

        float myY = Screen.height;
        myY = myY -130;

        if (GUI.Button (new Rect (30, myY, 100, 100), "Music", myStyle) )
        {
            if (MySingleton.Instance.myMusic)
                MySingleton.Instance.MusicOff();
            else
                MySingleton.Instance.MusicOn();

            MySingleton.Instance.myMusic = !MySingleton.Instance.myMusic;
        }
    }
}
```

MySingleton.CS

```
using UnityEngine;
using System.Collections;

public class MySingleton : MonoBehaviour {

    private static MySingleton instance;
    public static MySingleton Instance
    {
        get
        {
            if (instance == null)
            {
                GameObject mySingletonObject = new GameObject("MySingletonObject");
                DontDestroyOnLoad(mySingletonObject);
                instance = mySingletonObject.AddComponent<MySingleton>();
            }
            return instance;
        }
    }

    public bool changeLevel = false;
    public bool pause = false;
    public bool myMusic = true;

    void Awake()
    {
        AudioSource myMusic = gameObject.AddComponent<AudioSource> ();
        AudioClip myClip;
        myClip = (AudioClip)Resources.Load ("Sounds/Carefree");
        myMusic.clip = myClip;
        myMusic.loop = true;
        myMusic.Play ();
    }

    public void MusicOff()    {        audio.Stop ();    }

    public void MusicOn()
    {
        if (!audio.isPlaying)
            audio.Play ();
    }

}
```

PlantBlink .CS

```
public class PlantBlink : MonoBehaviour {

    public int myPlant = 4;
    private bool raise = false;

    void OnGUI() {
        GUI.color = Color.clear;
        float myY = Screen.height * (1-.57f) -110;

        if (GUI.Button(new Rect (30, myY, 100, 100), "") )
        {
            myPlant = 4;
            GameObject.Find ("TextTypeLevel").guiText.text = "Earth Level";
            GameObject.Find("TextExplanation").guiText.text = "Click on icons to select earth, water or light";
            ResetDimensions(6,280);
        }

        if (GUI.Button(new Rect (160, myY, 100, 100), "") )
        {
            myPlant = 5;
            GameObject.Find ("TextTypeLevel").guiText.text = "Water Level";
            GameObject.Find("TextExplanation").guiText.text = "Click on icons to select earth, water or light";

            ResetDimensions(4,30);
            ResetDimensions(6,280);
        }

        if (GUI.Button(new Rect (280, myY, 100, 100), "") )
        {
            myPlant = 6;
            GameObject.Find ("TextTypeLevel").guiText.text = "Light Level";
            GameObject.Find("TextExplanation").guiText.text = "Click on icons to select earth, water or light";
            ResetDimensions(4,30);
            ResetDimensions(5,160);
        }
    }

    void Update()
    {
        if (PlayerPrefs.GetInt ("Level") > 3)
        {
            GameObject goPlant = GameObject.Find("Plant"+myPlant);
```

```
            float myX = goPlant.guiTexture.pixelInset.x;
            float myY = goPlant.guiTexture.pixelInset.y;
            float myWidth = goPlant.guiTexture.pixelInset.width;
            float myHeight = goPlant.guiTexture.pixelInset.height;

            if(myWidth > 50 && !raise)
            {
                myWidth -= Time.deltaTime*50;
                myHeight -= Time.deltaTime*50;
                myX += Time.deltaTime * 25;
                myY += Time.deltaTime * 25;
            }
            else
                raise = true;

            if (myWidth < 100 && raise)
            {
                myWidth += Time.deltaTime*50;
                myHeight += Time.deltaTime*50;
                myX -= Time.deltaTime * 25;
                myY -= Time.deltaTime * 25;
            }
            else
                raise = false;

            goPlant.guiTexture.pixelInset = new Rect(myX,myY, myWidth,myHeight);

        }
    }

    void ResetDimensions(int i,int x)
    {
        GameObject goPlant = GameObject.Find("Plant"+i);
        goPlant.guiTexture.pixelInset = new Rect(x,0,100,120);
    }
}
```

PlayScript .CS

```
using UnityEngine;
using System.Collections;

public class PlayScript : MonoBehaviour {

    public GUIStyle myStyle = new GUIStyle();
    public GameObject myloadingGUI;


    void OnGUI()
    {

        float myX = Screen.width;
        myX = myX - 130;

        float myY = Screen.height;
        myY = myY-130;


        if (GUI.Button (new Rect (myX, myY, 100, 100), "Play", myStyle) )
        {
            if (PlayerPrefs.GetInt("Level")>3)
            {
                myloadingGUI.SetActive (true);
                GameObject myText = GameObject.Find("TextLoadHelp");

                PlantBlink goScript =(PlantBlink) GameObject.Find("MyGUI").GetComponent("PlantBlink");
                switch (goScript.myPlant)
                {
                    case 4:
                            MySingleton.Instance.MusicOff();
                            myText.guiText.text =     "Remember this is augmented reality!" +
                                                System.Environment.NewLine +
                                                "This level requires boths markers.";
                            Application.LoadLevel ("ARSimon");
                            break;
                    case 5:
                            MySingleton.Instance.MusicOff();
                            myText.guiText.text =     "This level requires plant marker.";
                            Application.LoadLevel ("ARwater");
                            break;
                    case 6:
                            myText.guiText.text =     "This level requires secundary marker.";
```

```
                    Application.LoadLevel ("ARLight");
                    break;
                }
            }
            else
            {
                MySingleton.Instance.MusicOff();
                Application.LoadLevel ("Tutorial");
            }
        }
    }
}
```

RemainScript.CS

```
using UnityEngine;
using System.Collections;

public class RemainScript : MonoBehaviour {

    void Awake() {
        DontDestroyOnLoad(transform.gameObject);
    }
}
```

ResetLevel .CS

```
using UnityEngine;
using System.Collections;

public class ResetLevel : MonoBehaviour {


    public int Level = 0;
    public int Earth = 0;
    public int Water = 0;
    // Use this for initialization
    void Start ()
    {
        PlayerPrefs.SetInt ("Level",Level);
        PlayerPrefs.SetInt ("Earth",Earth);
        PlayerPrefs.SetInt ("Water",Water);

    }
}
```

RotateScript .CS

```
using UnityEngine;
using System.Collections;

public class RotateScript : MonoBehaviour {

    public Transform rotateCenter;
    public float rps = 40f;
    public float radius = 250f;
    public float deltaRadius = 1f;

    // Update is called once per frame
    void Update () {
        if (!MySingleton.Instance.pause)
        {
            radius += deltaRadius * Time.deltaTime;
            transform.position = new Vector3 (rotateCenter.position.x, rotateCenter.position.y + 300, rotateCe
nter.position.z);
            transform.Translate (radius, 0, 0);
            transform.RotateAround (rotateCenter.position, Vector3.up, rps * Time.deltaTime);
        }
    }
}
```

ScreenSkin .CS

```
using UnityEngine;
using System.Collections;
using System;
using System.IO;

public class ScreenSkin : MonoBehaviour
{


    private bool torchMode = false;

    private Texture btnFlashTexture;
    private Texture btnCamera;
    private Texture btnAchievements;
    public Transform locateAchievements;
    public Transform locateScore;

    private void Start()
    {
        Screen.orientation = ScreenOrientation.Landscape;
        CameraDevice.Instance.SetFocusMode(CameraDevice.FocusMode.FOCUS_MODE_TRIGGERAUT
O);

        btnCamera = (Texture) Resources.Load ("Images/camera");
        btnAchievements = (Texture) Resources.Load ("Images/help");
        Time.captureFramerate = 25;
        MySingleton.Instance.pause = false;
        locateAchievements.Translate(Vector3.up * 2f);
    }


    private void Update()
    {

        if (Input.GetKeyDown(KeyCode.Escape))
            Application.LoadLevel(1);
    }

    void OnGUI ()
    {

        if (MySingleton.Instance.pause)
        {
```

```
    GUI.backgroundColor = new Color (0.33F, 0.42F, 0.18F);
    // texts inside GameObjectEmpty LocateHelp
    GameObject.Find("LocateHelp");
    if (GUI.Button (new Rect (12, 12, Screen.width - 24, Screen.height - 24), ""))
    {
          locateAchievements.Translate(Vector3.up * 2f);
          locateScore.Translate(Vector3.down * 2f);
          MySingleton.Instance.pause = false;
    }
}
else
{
    GUI.backgroundColor = Color.clear;

    //Flashlight's Button
    if (torchMode)
       btnFlashTexture = (Texture)Resources.Load ("Images/light_bulb_on");
    else
       btnFlashTexture = (Texture)Resources.Load ("Images/light_bulb_off");

    if (GUI.Button (new Rect (0, 0, 128, 128), btnFlashTexture))
    {
       torchMode = !torchMode;
       CameraDevice.Instance.SetFlashTorchMode (torchMode);
    }


    //Photo's Button

    if (GUI.Button (new Rect (Screen.width - 128, 0, 128, 128), btnCamera))
    {
       string filePhoto;
       if (Application.platform == RuntimePlatform.Android)
          filePhoto = AndroidPhotoStore();
       else
          filePhoto = MACPhotoStore();


       Texture2D tex = new Texture2D(Screen.width, Screen.height,TextureFormat.RGB24,true);
       tex.ReadPixels(new Rect(0,0,Screen.width,Screen.height),0,0);
       tex.Apply ();
       byte[] bytes = tex.EncodeToPNG();
       System.IO.File.WriteAllBytes(filePhoto, bytes);
    }
```

```
            //help's button
            if (GUI.Button (new Rect (0, Screen.height-128, 128, 128), btnAchievements))
            {
                locateAchievements.Translate(Vector3.down * 2f);
                locateScore.Translate(Vector3.up * 2f);
                MySingleton.Instance.pause = true;
            }
        }
    }

    string MACPhotoStore()
    {
        string theTime = System.DateTime.Now.ToString("hhmmss");
        string theDate = System.DateTime.Now.ToString("MMddyyyy");
        string destination = "/users/Chus/Desktop/PotPlants" + theDate + theTime + ".png" ;
        return destination;
    }

    string AndroidPhotoStore()
    {
        //set name of the picture as PNG
        using (AndroidJavaObject jo = new AndroidJavaObject("android.os.Environment"))
        {
            //get directory on android systems...
            AndroidJavaObject externalStorageDirectory = jo.CallStatic<AndroidJavaObject>("getExternalStor
ageDirectory");
            string cameraDir = externalStorageDirectory.Call<string>("getPath") + "/DCIM/Camera/";

            //set Filename
            string fileName = System.DateTime.Now.ToString("MMddyyyy_")  + System.DateTime.Now.ToStri
ng("hhmmss") + ".png" ;

            string origin = cameraDir + fileName;
            return origin;

        }
    }

}
```

ShotScript .CS

```
using UnityEngine;
using System.Collections;

public class ShotScript : MonoBehaviour {

    public Transform shotPref;
    public float coolDown = 0.5f;
    public GameObject rain;
    public AudioClip thunder,winner;

    private float nextShot = 0f;
    public GUIText myScore,myRecord,myRemain;
    public int score,remainingClouds;
    public bool scored = false;
    private bool wining = false;


    void Start ()
    {
        score = 0;
        myScore.guiText.text = "0";
        remainingClouds = PlayerPrefs.GetInt ("Water");
        myRecord.guiText.text = remainingClouds.ToString();

        if (remainingClouds > 0)
                remainingClouds = 1 + (remainingClouds / 10);
        else
                remainingClouds = 3;
        myRemain.text = remainingClouds.ToString ();
    }


    void LoadLevelsScene()
    {
        Application.LoadLevel ("Levels");
    }

    void Update ()
    {
        if (!MySingleton.Instance.pause && !wining)
        {
            if ((Input.GetButtonDown ("Fire1") || Input.touchCount > 0) && Time.time > nextShot)
            {
```

```
            nextShot = Time.time + coolDown;
            Instantiate (shotPref, transform.position, transform.rotation);
        }
    }

    if (scored)
    {
        GameObject.Find ("CloudSpawn").GetComponent<RotateScript>().rps = (int) Random.Range(20,
100);
        GameObject.Find ("CloudSpawn").GetComponent<RotateScript>().radius = (int) Random.Range(
250,500);
        GameObject.Find ("CloudSpawn").GetComponent<RotateScript>().deltaRadius = (int) Random.R
ange(1,10);
        audio.PlayOneShot(thunder);
        scored = false;
        score += 10;
        remainingClouds --;

        myScore.guiText.text = score.ToString();
        myRemain.text = remainingClouds.ToString ();

        if (remainingClouds == 0)
        {
            audio.PlayOneShot(winner);
            PlayerPrefs.SetInt ("Water", score);
            MySingleton.Instance.changeLevel = true;
            rain.SetActive(true);
            wining = true;
            Invoke("LoadLevelsScene",10);
        }

    }
  }
}
```

simonColor .CS

```
using UnityEngine;
using System.Collections;

public class simonColor : MonoBehaviour {


    public Transform myColorbutton;
    public float myDist;
    // Update is called once per frame
    void Update () {

        GameObject seed = GameObject.Find("Seed");
        if (seed.GetComponent<SymonThread> ().gameStat == SymonThread.State.Reply)
        {
            myDist = Vector3.Distance (gameObject.transform.position, myColorbutton.position);
            Color myColor = gameObject.renderer.material.color;
            myColor.a = 0f;
            if (myDist < 450 && GameObject.Find ("UNDER").renderer.enabled && GameObject.F
ind("TextStart").renderer.enabled)
                    myColor.a = 1f;
            gameObject.renderer.material.color = myColor;
        }
    }
}
```

soundColor .CS

```
using UnityEngine;
using System.Collections;

public class soundColor : MonoBehaviour {

    // Update is called once per frame
    public AudioClip simonSound;
    private bool playing = false;

    void Update ()
    {
        if (gameObject.renderer.material.color.a == 0)
                    playing = false;
        else
            if (gameObject.renderer.material.color.a == 1 && !playing)
            {
                    audio.PlayOneShot (simonSound);
                    playing = true;
            }
    }
}
```

SymonThread.CS

```
using UnityEngine;
using System.Collections;

public class SymonThread : MonoBehaviour {

    public GameObject myButtons;
    private string[] btn = {"SimonRed", "SimonGreen", "SimonBlue", "SimonYellow"};
    public enum State {Create, Reproduce, Reply, Off, Win, Loose};
    public State gameStat;
    public bool release;
    public int buttonPressed = 5;
    public GUIText myScore,myRecord;
    public GameObject btnVirtual;
    public GameObject mySimon;

    public AudioClip winner;
    public AudioClip looser;

    private int score;
    public int[] mySequence, rplySequence;
    private int maxElements;
    private int counter;

    private float delay;
    private const float MAXDELAY = 2f;
    private const float MINDELAY = .5f;
    public bool waitDelay = false;

    void Start ()
    {
        gameStat = State.Off;
        score = 0;
        myScore.guiText.text = score.ToString ();
        myRecord.guiText.text = PlayerPrefs.GetInt("Earth").ToString();
    }

    void Update () {

        if (!MySingleton.Instance.pause)
        {
                switch (gameStat)
                {
                        case State.Create:
```

```
                CreateSequence ();
                break;
            case State.Reproduce:
                ShowSequence ();
                break;
            case State.Reply:
                if (!myButtons.activeSelf)
                    myButtons.SetActive (true);
                GetSequence ();
                break;
            case State.Loose:
                if (mySimon.activeSelf)
                    ShowSequence ();
                break;
        }
    }
}

void CreateSequence()
{
    if (PlayerPrefs.GetInt ("Level") == 1)
        maxElements = 2;
    else
        maxElements = (PlayerPrefs.GetInt ("Earth")/10)+ 1;

    mySequence = new int[maxElements];
    rplySequence = new int[maxElements];
    for (int i = 0; i<maxElements; i++)
    {
        mySequence [i] = (int) Random.Range (0, 3.99f);
        rplySequence [i] = 5;
    }

    //initialize reproduce State
    counter = 0;
    delay = MAXDELAY;

    gameStat = State.Reproduce;
}

void ShowSequence()
{

    GameObject[] gos;
    gos = GameObject.FindGameObjectsWithTag ("ButtonColors");
```

```
if (waitDelay)
{
    if (delay > 0)
            delay -= Time.deltaTime;
        else
        {
            delay = MAXDELAY;
            waitDelay = false;
        }
}
else
{
    if (delay > 0)
        delay -= Time.deltaTime;
    else
    {
        delay = MINDELAY;
        waitDelay = true;
        if (counter < maxElements - 1)
            counter ++;
        else
        {
            //initialize raply State
            counter = 0;
            if (gameStat == State.Reproduce)
            {
                gameStat = State.Reply;
                GameObject.Find("TextStart").GetComponent<TextMesh>().text =     "Move Me" +
                                                                    System.Environment.NewLine +
                                                                    "to Colors";
            }
            if (gameStat == State.Loose)
                SwitchToButton();
        }
    }

    foreach (GameObject go in gos)
    {
        Color myColor = go.renderer.material.color;
        myColor.a = .0f;
        if (go.name == btn [mySequence [counter]] && !waitDelay)
            myColor.a = 1;
        go.renderer.material.color = myColor;
    }
}
```

```
}

void LoadLevelsScene()
{
    Application.LoadLevel ("Levels");
}

void GetSequence()
{

    if (release)
    {
        if (rplySequence [counter] == 5)
        {
            rplySequence [counter] = buttonPressed;
            if (rplySequence[counter] != mySequence[counter])
            {
                counter = 0;
                SwitchToButton();
                audio.PlayOneShot(looser);
                btnVirtual.SetActive(true);
                gameStat = State.Loose;
            }
        }

    }
    else
    {
        if (rplySequence [counter] != 5)
        {
            counter ++;
            score += 10;
            myScore.guiText.text = score.ToString ();
        }
    }

    if (counter == maxElements)
    {
        GameObject.Find("TextStart").GetComponent<TextMesh>().text =     "Good Job!";
        PlayerPrefs.SetInt ("Earth", score);
        MySingleton.Instance.changeLevel = true;
        audio.PlayOneShot(winner);
        Invoke("LoadLevelsScene",5);
        gameStat = State.Win;
    }
```

```
    }

    void SwitchToButton()
    {
        waitDelay = false;
        delay = MAXDELAY;
        mySimon.SetActive (false);
        GameObject.Find("TextStart").GetComponent<TextMesh>().text = "Solution";
    }

    public void reDoSequenze()
    {
        mySimon.SetActive (true);
    }
}
```

TextExplanationScript .CS

```
using UnityEngine;
using System.Collections;

public class TextExplanationScript : MonoBehaviour {

    string [] myText = new string[3];
    void Start ()    {
        myText [0] =      "The soil of the plant contains 4 elements:" +
                    System.Environment.NewLine +
                    "substrate, minerals, vitamins and organic" +
                    System.Environment.NewLine +
                      "matter." +
                    System.Environment.NewLine +
                    "Remember the sequence of colors.";

        myText [1] =      "The plant needs water to perform" +
                    System.Environment.NewLine +
                    "photosynthesis." +
                    System.Environment.NewLine +
                    "Usually the rain gives all the necessary.";

        myText [2] =      "Also the plant needs light to complete" +
                    System.Environment.NewLine +
                    "photosynthesis:" +
                    System.Environment.NewLine +
                    "Usually the Sunlight gives it.";
    }
    void Update () {
        switch (PlayerPrefs.GetInt ("Level"))
        {
            case 1:
                gameObject.guiText.text = myText[0];
                break;

            case 2:
                gameObject.guiText.text = myText[1];
                break;
            case 3:
                gameObject.guiText.text = myText[2];
                break;
        }
    }
}
```

TextTypeLevel.CS

```
using UnityEngine;
using System.Collections;

public class TextTypeLevel : MonoBehaviour {


	void Update ()
	{
	   switch (PlayerPrefs.GetInt ("Level"))
	   {
	   case 1:
	      gameObject.guiText.text = "Earth Tutorial";
	      break;

	   case 2:
	      gameObject.guiText.text = "Water Tutorial";
	      break;
	   case 3:
	      gameObject.guiText.text = "Light Tutorial";
	      break;
	   }
	}
}
```

Tutorial .cs

```
using UnityEngine;
using System.Collections;

public class Tutorial : MonoBehaviour {

    // Use this for initialization
    private int page = 0;
    private int myLevel;
    public GUIText myText;
    public GUITexture myTexture;
    public GUIStyle myStyle = new GUIStyle();

    void Start () {

        Screen.orientation = ScreenOrientation.Landscape;

        myLevel = PlayerPrefs.GetInt ("Level");
        switch (myLevel)
        {
            case 1:
                myText.text =     "The plant cannot be grown just anywhere." +
                                  System.Environment.NewLine +
                                  "Make the best soil for your plant." +
                                  System.Environment.NewLine +
                                  "All the plants need good soil in order" +
                                  System.Environment.NewLine +
                                  "to grow up strong and healthy.";

            break;
            case 2:

                myText.text =     "The rain water is better for the plants" +
                                  System.Environment.NewLine +
                                  "than tap water." +
                                  System.Environment.NewLine +
                                  "The plant needs large quantities of water for growth.";
            break;
            case 3:
                myText.text =     "Grow lights are designed to stimulate"+
                                  System.Environment.NewLine +
                                  "the plant growth" +
                                  System.Environment.NewLine +
                                  "But the lamp is broken." +
```

```
                        System.Environment.NewLine +
                        "Repair it completing the maze.";
        break;
    }
}

// Update is called once per frame
void OnGUI()
{
    float myX = Screen.width;
    myX = myX - 130;

    float myY = Screen.height;
    myY = myY -130;

    if (Input.GetKeyDown(KeyCode.Escape))
        Application.LoadLevel("Levels");


    if (GUI.Button (new Rect (myX, myY, 100, 100), "Next",myStyle ))
    {
        page ++;
    }

    switch (myLevel)
    {
    case 1:
            if (page== 1)
            {
                    myText.text =     "Use secundary marker";
                    myTexture.texture = Resources.Load("Images/e1") as Texture;
            }
            if (page== 2)
            {
                    myText.text =     "Place your hand over the marker to start"+
                            System.Environment.NewLine +
                            "Remember, nothing happens if you touch the screen";
                    myTexture.texture = Resources.Load("Images/e2") as Texture;
            }
            if (page== 3)
            {
                    myText.text =     "See the sequence of colors in the marker";
                    myTexture.texture = Resources.Load("Images/e3") as Texture;
            }
            if (page== 4)
```

```
        {
                myText.text =     "Using both markers, move the secondary" +
                        System.Environment.NewLine +
                        "to color areas.";
                myTexture.texture = Resources.Load("Images/e4") as Texture;
        }
        if (page== 5)
        {
                myText.text =     "Loading...";
                Application.LoadLevel ("ARSimon");
        }
    break;
case 2:
        if (page== 1)
        {
                myText.text =     "Use primary marker";
                myTexture.texture = Resources.Load("Images/w1") as Texture;
        }
        if (page== 2)
        {
                myText.text =     "Seek clouds and fires it." +
                        System.Environment.NewLine +
                        "To shot your ammo touch the screen.";
        }
        if (page== 3)
        {
                myText.text =     "Loading...";
                Application.LoadLevel ("ARWater");
        }
    break;
case 3:
        if (page== 1)
        {
                myText.text =     "Use secundary marker";
                myTexture.texture = Resources.Load("Images/l1") as Texture;
        }
        if (page== 2)
        {
                myText.text =     "Turns around the marker" +
                        System.Environment.NewLine +
                        "to move the ball";
        }
        if (page== 3)
        {
                myText.text =     "Loading...";
```

```
                Application.LoadLevel ("ARLight");
            }
            break;
        }
    }
}
```